NAVAL POSTGRADUATE SCHOOL

Monterey, California

THESIS

SHIFT AND SCALE INVARIANT PREPROCESSOR

by

Norman E. Huston, Jr.

December 1981

Thesis Advisor: L. A. Wilson

Approved for public release; distribution unlimited

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS
BEFORE COMPLETING FORM

1. REPORT NUMBER

2. GOVT ACCESSION NO.

3. RECIPIENT'S CATALOG NUMBER

4. TITLE (and Subtitle)
Shift and Scale Invariant Preprocessor

5. TYPE OF REPORT & PERIOD COVERED
Engineer's thesis, December 1981

6. PERFORMING ORG. REPORT NUMBER

7. AUTHOR(s)
Norman Earl Huston, Jr.

8. CONTRACT OR GRANT NUMBER(s)

9. PERFORMING ORGANIZATION NAME AND ADDRESS
Naval Postgraduate School
Monterey, California 93940

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS

11. CONTROLLING OFFICE NAME AND ADDRESS
Naval Postgraduate School
Monterey, California 93940

12. REPORT DATE
December 1981

13. NUMBER OF PAGES
134

14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)

15. SECURITY CLASS. (of this report)
Unclassified

15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release, distribution unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

shift invariance, scale invariance, Mellin transform, classification preprocessor, range only radar classification

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

A preprocessor is designed to extract a set of features that enhance natural clustering by removing extraneous information. The design removes time shift and scale dependence by taking advantage of invariant properties of a Fourier transform followed by a Mellin transform. The preprocessor is realized using an FFT and a Mellin transform with a conventional

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-014-6601

error correction term. The error term proves to be indeterminate, but the error's bound is identified as the envelope for Mellin correction terms. Properties of the Mellin transform are employed to modify the signal so that the error correcting is no longer required. The resulting algorithms are tested with variously scaled inputs for which closed form solutions are known. With a verified modification in place, the preprocessor produces features that are invariant to shifting and scaling, while retaining enough information to classify canonic shapes. A method of improving performance is introduced.

Shift and Scale Invariant Preprocessor

by

Norman E. Euston, Jr.
Lieutenant Commander, United States Navy
B.S.E.E. and B.N.S., University of Wisconsin, 1971

Submitted in partial fulfillment of the
requirements of the degrees of

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING
and
ELECTRICAL ENGINEER

from the

NAVAL POSTGRADUATE SCHOOL
December 1981

ABSTRACT

A preprocessor is designed to extract a set of features that enhance natural clustering by removing extraneous information. The design removes time shift and scale dependence by taking advantage of invariant properties of a Fourier transform followed by a Mellin transform. The preprocessor is realized using an FFT and a Mellin transform with a conventional error correction term. The error term proves to be indeterminate, but the error's bound is identified as the envelope for Mellin correction terms. Properties of the Mellin transform are employed to modify the signal so that the error correcting is no longer required. The resulting algorithms are tested with variously scaled inputs for which closed form solutions are known. With a verified modification in place, the preprocessor produces features that are invariant to shifting and scaling, while retaining enough information to classify canonic shapes. A method of improving performance is introduced.

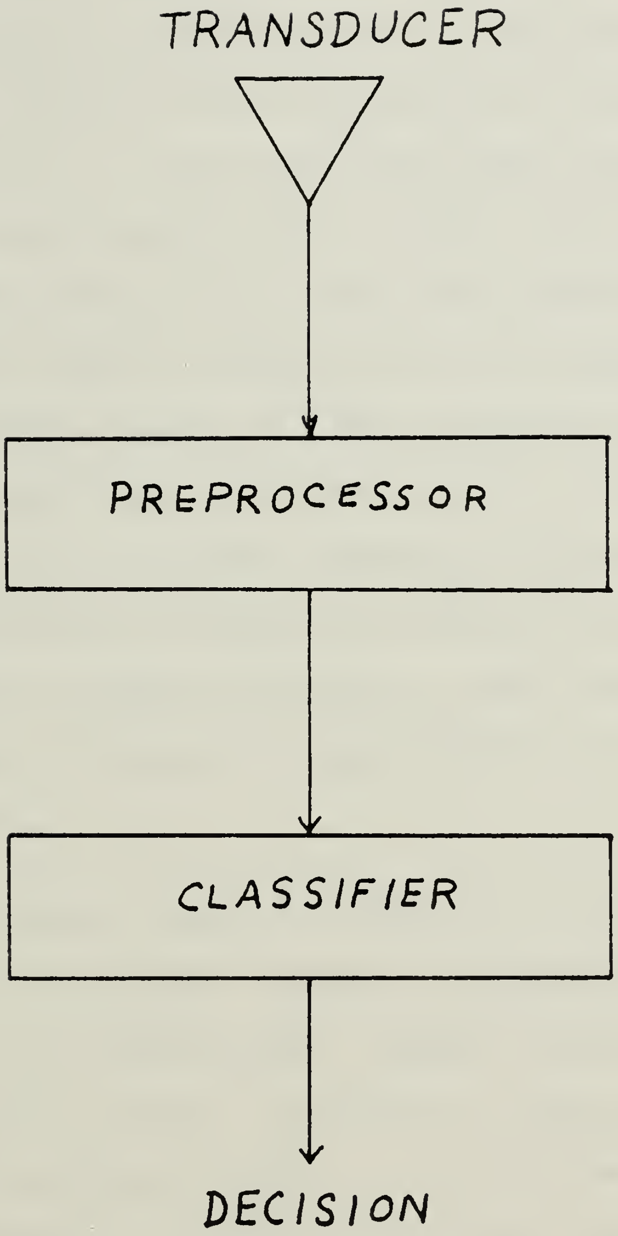
TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

I. INTRODUCTION AND BACKGROUND

Pattern classification is the assignment of a physical object or event to one of several prespecified catagories and is the result of an incomplete theory of perception. Although many transducers are available for converting light, sound, temperature, reflected radar signals, etc., to electrical signals, the ability of machines to perceive or to recognize their environment remains very limited. In the structured world of communcations engineering, signals are designed to be detectable and differentiable. A much more difficult problem presents itself when sensing an environment through a transducer and recognizing or even classifying the elements of that environment on the sensed characteristics of the transducer's electrical output. Pattern recognition can be considered a complex communications problem (for example, attempting to teach a machine to decode signals encoded by nature). It is possible to alter the transducer's output to facilitate object classification, but determining how to alter that output is not a simple task. The main elememts of a classification system is shown in figure 1 [1]. The transducer senses, actively or passively, a set of characteristics belonging to the object. These characteristics can never be a complete description of an



A Classification System

Figure 1

object, but represent, hopefully, enough information to classify the object as belonging to one of a number of classes. For instance, temperature is a characteristic of the object and a class, but this feature is of little value unless it differs in some way from objects belonging to other classes, while the set must include characteristics that are common among that class. The preprocessor (or feature extractor) aims to reduce the data by measuring or quantifying certain properties that distinguish the sensed object as belonging to one class and not to others. This can be done by discerning key features or using the imput to generate another set of features optimized by some rule. The values of each of these features is then passed to the classifier, which evaluates these features to assign the object to a class.

With varied success, machine pattern classification has been applied to a large range of problems/disciplines. Fields where it is particularly common include optical imagery, acoustic signal processing, radiology, radio astronomy, and electronic warfare, to name a few. Work in many of these fields was reviewed during the development of this thesis and the results derived and demonstrated here may, in turn, be applicable to the field in general. This effort has been directed toward designing a preprocessor to produce a set of features that are invariant to information

known to be superfluous to classification, but that retain enough information to classify an object. The object has been sensed by a transducer and has been represented as an empirically derived, univariate time series. Such a series would be the form of data available from range only radar return which is specifically what the preprocessor is designed to handle. Returning to figure 1, the object has an infinite set of characteristics (here portrayed as an infinite series of discrete values). The transducer/receiver has collected some characteristics of the target objective in the presence of noise. This information is relayed as a set of discrete samples (hi) from a band limited signal. The preprocessor is designed to determine and code revelent information (Hj) for the classifier. If this task was done well, classification becomes a trivial problem. On the other hand, if the classifier becomes ideal (capable of resolving an infinite number of characteristics in noise) the preprocessor design begins to look like a wire. The distinction between the preprocessor and the classifier is arbitrary from an analytical point of view. When designing a classification system functionally, a difference is enforced. The classifier has little concern for how the features are developed, but seeks to efficiently use those provided to guess the class of the target object. The preprocessor is

problem dependent, needing to produce an optimal set of features, Ej, from the sensed data hi.

A. FEATURE EXTRACTION

For the purposes of this paper, two generic approaches to feature extraction are defined. The first, a classification approach, was described above. The second, a descriptive approach, tries to define the object in terms of the objects' structural features. This system might recognize a car, for example, by breaking up the visual picture into canonic shapes, and comparing this to previously specified canonic class models. The perceived structure of the physical object is maintained and should reflect the structure of the object itself. This approach could be robust to temporary changes in the object itself. In the car example, knowing that at one end of the car the trunk can be opened or closed allows the device to take this factor into account. Another important advantage to descriptive techniques is that the class characteristics may be entered or specified without collecting actual transducer generated data to train the machine. Unfortunately, the problem of designing a machine to analyse a visual scene to produce a structural description has proved to be quite difficult. Object description from a univariate time series is even more difficult, and if the

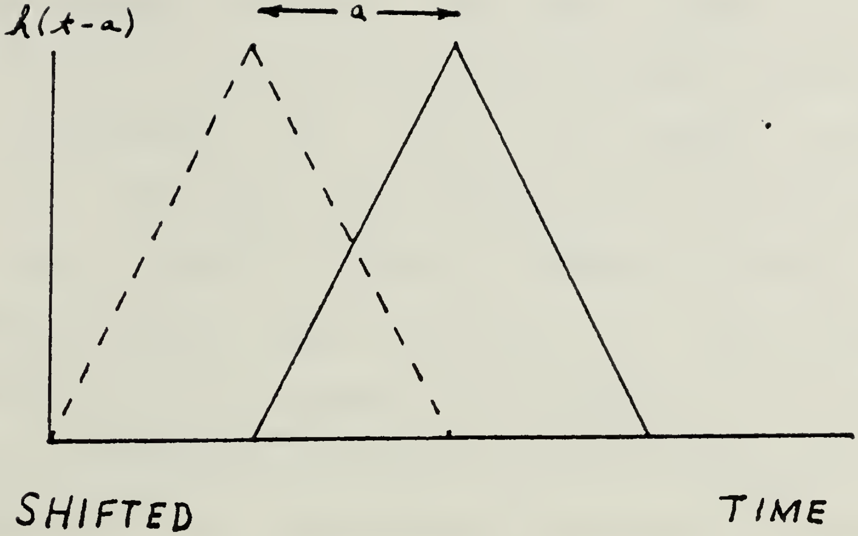
radiation sensed by the transducer is not from the visual spectrum, the task rapidly approaches the impossible. For these reasons the approach taken was the classification approach (to reduce the signal to a set of orthogonal features that do not uniquely reflect the structure of the object, but do retain sufficient information to classify the object).

This paper excludes a detailed description of the transducer specification. The problem of the classifier itself is viewed as one of partitioning the feature space (Ej) into regions; one region for each category. Ideally, this partitioning should be arranged so that none of the decisions are ever wrong. When this cannot be realized, at least the probability of error should be minimized or the average cost of errors minimized. The problem is one within statistical decision theory. Knowledge of the object classes (the transducer and the classifier) are required to design the preprocessor, which is the topic of this thesis. The preprocessor designed and built here generates features from a range only radar video signal. These features are used by a general Bayesian learning classifier. The supervised learning general Bayesian classifier approaches the problem by taking a series of incoming sets of features labeled as to their class. From the data, an a posteriori density is computed. Each successive set of training data

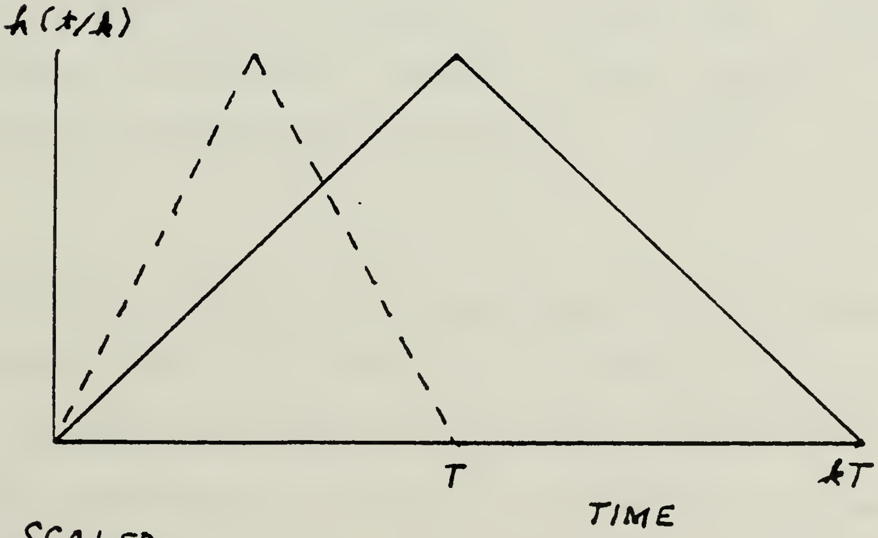
is used to refine the densities′ statistics. When the classifier has been trained on N classes, the features are modified to separate the class volumes in an optimal way and to reduce the number of features to one less than the number of possible classes. The feature vectors of class members are clustered about a simplex point and likely boundaries are set up allowing classification of the object as belonging to one of the classes, or of an unkown class. An N simplex is a collection of N points in (N-1) space where the distance between any two of the points is equal to the distance between any other two. Thus a three class problem transformed into a three simplex in a two dimensional plane produces clustering of the three class′s about the vertices of an equilateral triangle. The simplex coordinates are the reduced feature vectors, generalized from the training data. In a controlled, simple problem the classifier works well, but when encountering real problems one class′s feature space will intertwine another′s, making it much more difficult to obtain separation in a meaningful way. The goal of the preprocessor is to present key features that determine class for subsequent optimization by the classifier.

B. FOURIER - MELLIN PREPROCESSING

Common to all univariate, time series classification problems are several variables that interfere with the recognition process. Assuming discrete data processing is used, these are addressed in the following order; windowing, framing, scaling, sampling rate, quantization noise, and sufficient information. For real processing, the input waveform is not sampled for all time. It is sampled for a period of time. This windowing of the data corrupts the resulting spectrum in two ways [2]. First it introduces a periodicity (the inverse of the window length) and resulting aliasing to the otherwise infinite spectrum, and further distorts the spectrum by a convolution with the spectrum of the window itself. Both of these effects will color all of the data in the same way and so can be accounted for by deterministic methods. Framing can be considered characteristic of poor synchronization, whereby the pattern of concern is not position stable with repect to the window as shown in figure 2a. It seems that even in human optical recognition, the eye tends to center the pattern prior to recognition, unless trained otherwise. Scaling is that property whereby the object field may vary in scale or aspect angle, in one dimension or several dimensions as in figure 2b. Before the analog signal is sampled, it must be fed through a sharp low pass filter,



a. SHIFTED



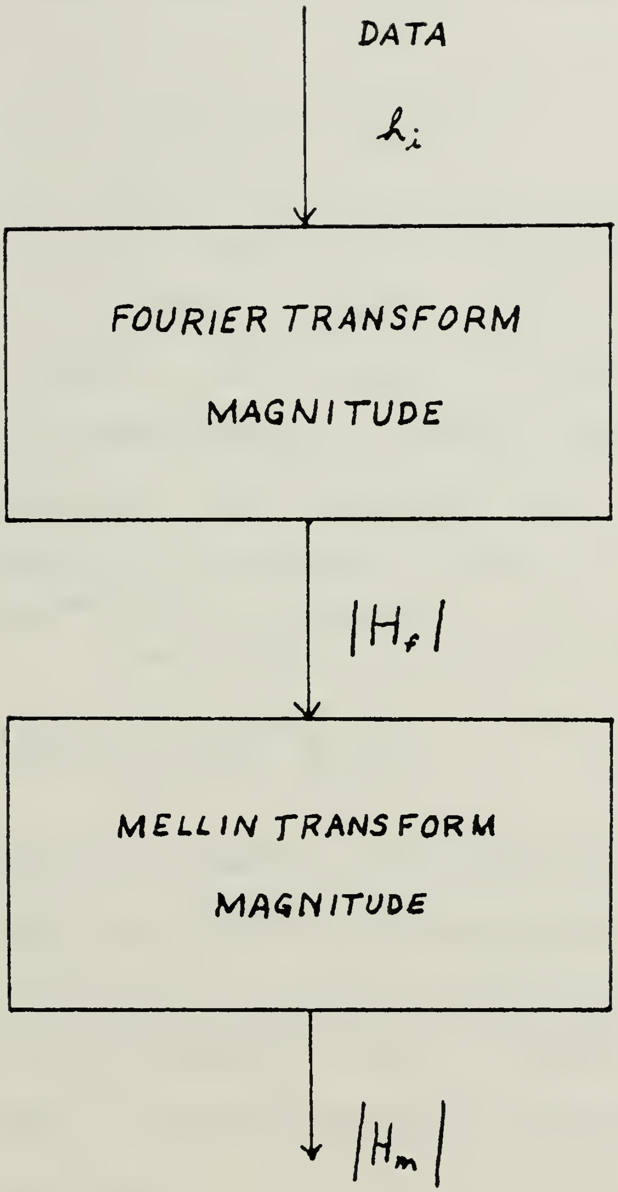
b. SCALED

Shifting and Scaling Variation

Figure 2

because no higher frequency noise can be present without being folded onto the valid data. Quantization noise, due to the requirement to round off each sample to some discrete level, is treated the same as round off error in numerical processing [3]. In all of these problems discussed to this point, the effect of this processing is to mask the actual feature vectors, making the classification system less sensitive to valid pattern characteristics. In all recognition problems, it is assumed that there is sufficient information present for a pattern to be detected and identified by the system. This means that there is sufficient variability between the classes, but sufficient similarity between those patterns of a single class to classify each pattern in terms of those classes.

A verifiable goal of this preprocessor is to produce a set of features that are invariant to shifting and scaling changes. An approach, figure 3, has been proposed and used [4-10]. The preprocessor consists of putting the sampled data successively through two transforms, a fast Fourier transform and a discrete Mellin transform. Integral transforms, the Fourier and Mellin included, develop naturally from the solution of simple problems in potential



An FM Preprocessor

Figure 3

theory [11-13]. The Fourier integral transform of the waveform h(t),

$$H(f) = \int_{-\infty}^{\infty} h(t)\, K(f,t)\, dt \tag{1}$$

where $K=\exp[-2\pi ft]$, is a principle analytical tool in such diverse fields as linear systems, optics, probability theory, quantum physics, and signal analysis [13]. Its purpose in the preprocessor is twofold, but relies on a single characteristic. The magnitude of the Fourier transform is invariant to shifting, h(t-a).

$$\mathcal{F}[h(t-a)] = e^{-j2\pi fa}\int_{-\infty}^{\infty} h(t)\, e^{-j2\pi tf}\, dt = e^{sa} H(s)$$

$$|H(s)| = |e^{sa} H(s)| \tag{2}$$

This characteristic removes the effects of framing inaccuracies, and also permits the averaging of successive looks or pulses of data to improve feature resolution, but removes much of the information about a signal´s structure as discussed later. A discrete Fourier transform,

$$\begin{aligned} h(t) &\rightarrow h(m\Delta) \qquad && m = 0, 1, 2, \ldots, M-1 \\ H(f) &\rightarrow H(n/T) \qquad && n = 0, 1, 2, \ldots, N-1 \end{aligned} \tag{3}$$

$$H(n) = \sum_{m=0}^{M-1} h(m)\, e^{-j2\pi mn/N} \tag{4}$$

has an equivalent identity, but it is only exact for shifts of integer values.

$$|\mathcal{F}_D[h(m-i)]|$$

$$= \left| H(m)\, e^{-j2\pi i m/N} \right| = |H(m)| \qquad (5)$$

Shifts of other than integer values result in errors that depend not only on the shift, but on qualities of the sampled waveform itself, h(t).

The Mellin transform is an integral transform with the kernel, K.

$$K = t^{s-1} \qquad (6)$$

$$H(s) = \int_0^{\infty} h(t)\, t^{s-1}\, dt \qquad (7)$$

is the Mellin transform with respect to the complex parameter $s=r-j2\pi m$. Several simple substitutions relate this to more common analytical tools. Exponentially warping t=exp[x] changes the appearance of the integral to what is often called a double sided Laplace transform [14],

$$H(s) = \int_{-\infty}^{\infty} h(e^{x})\, e^{sx}\, dx \qquad (8)$$

The transform is invariant to t domain scaling when taken with respect to the imaginary part alone,

$$H(s) = \int_{-\infty}^{\infty} h'(x)\, e^{-j2\pi m x}\, dx \qquad (9)$$

Equation (9) is recognized as the Fourier intergral of an exponentially distorted waveform $h'(x)$. The modulus of the expression on the right is the magnitude of the Fourier transform of the exponentially distorted time function. The property to be exploited in a Fourier-Mellin (FM) preprocessor is that the modulus of the transform in s, is invariant to t-scaling. Given a time waveform h(t), its Mellin transform H(s) is given as equation (7). Scaling the entire t-domain by k,

$$\mathcal{M}[h(t/k)] = \int_0^\infty h(t/k)\, t^{s-1}\, dt \tag{10}$$

Letting r=t/k, and remembering that s is imaginery,

$$k^s \int_0^\infty h(r)\, r^{s-1}\, dr = k^s\, H(s)$$

$$|k^s H(s)| = |H(s)| \tag{11}$$

Much effort and detail is spent implementing equation (7) digitally in Chapters II and III. The rest of this chapter is devoted to providing some required background on the discrete versions of the Fourier transform and some properties upon which the FM preprocessor depends. The treatment here is brief, being mainly a review of basic FFT concepts and as such may be skipped without loss of content.

A discrete Fourier transform is not computationally efficient and so leads to impractically long processing

times. The fast Fourier transform (FFT) efficiently computes the discrete transform and is used in the preprocessors built for this thesis. Other properities of the FFT should be presented before getting into the detailed design of the preprocessor. The first and last are concerned with symmetry. If h(m) is real, as in the case of the sampled data, then the frequency spectrum of that data is even, |He(n)|=|He(-n)|. H(n) has a real part and an imaginary part, Re(n) and Im(n), while h(m) has an even he(m)=he(-m), and an odd ho(m)=-ho(-m) part.

$$Re(m) = \sum_{m=0}^{N-1} he(m) \cos\left(\frac{2\pi n m}{N}\right)$$

$$Im(m) = \sum_{m=0}^{N-1} h_o(m) \sin\left(\frac{2\pi n m}{N}\right) \qquad (12)$$

The odd part of h(n) times the cosine kernel summation, and the even part of h(n) times the sine kernel summation are both zero. H(n) can now be seen to have an even and odd part. Taking the magnitude of an odd function makes it even, proving that the Fourier transform of a real series is a spectrum whose magnitude is symmetric about f=0. This is of course true for both the integral and discrete Fourier transforms. The second line of symmetry is an effect of discretizing the signal and its spectrum for a discrete transform as evinced by its theoretical development. Before proceeding though, the convolution theorem is required.

The convolution of the two functions h(t) and g(t) is defined as the familiar integral,

$$y(t) = \int_{-\infty}^{\infty} h(\tau)\, g(t-\tau)\, d\tau = h(t) * g(t)$$

$$= \int_{-\infty}^{\infty} g(\tau)\, h(t-\tau)\, d\tau = g(t) * h(t) \qquad (13)$$

The relationship between convolution and the Fourier integral is very important to modern analysis and contributes to making the Fourier transform a key analytic tool. The convolution theorem states that if h(t) has a Fourier transform F(f) and g(t) has the Fourier transform G(f), then h(t)*g(t) has the Fourier transform H(f)G(f).

$$y(t) = h(t) * g(t)$$

$$\mathcal{F}[y(t)] = H(f)\, G(f) \qquad (14)$$

Proving this, the Fourier integral is used directly on the convolution integral.

$$Y(f) = \int_{-\infty}^{\infty} \left[\int_{-\infty}^{\infty} g(\tau)\, h(t-\tau)\, d\tau \right] e^{-j2\pi f t}\, dt$$

$$= \int_{-\infty}^{\infty} g(\tau) \left[\int_{-\infty}^{\infty} h(t-\tau)\, e^{-j2\pi f t}\, dt \right] d\tau \qquad (15)$$

From the shifting theorem already presented,

$$Y(f) = \int_{-\infty}^{\infty} g(\tau) \left[e^{-j2\pi f \tau} H(f) \right] d\tau$$

$$= H(f) \int_{-\infty}^{\infty} g(\tau)\, e^{-j2\pi f \tau}\, d\tau \qquad (16)$$

And finally,

$$Y(f) = \mathcal{F}[g(t) * h(t)] = G(f)H(f) \qquad (17)$$

It can be shown similarly that,

$$\mathcal{F}^{-1}[G(f) * H(f)] = g(t)h(t) \qquad (18)$$

Clearly, convolution in one domain is simple multiplication in the other domain. Although not needed for the pending development of the discrete Fourier transform, another important relationship, known as the correlation theorem, can be appropriately dealt with here. The correlation integral,

$$z(t) = \int_{-\infty}^{\infty} h(\tau)\, g(\tau + t)\, d\tau \qquad (19)$$

has an operation with which it forms a Fourier pair as did the convolution-multiplication operations. This theorem can be established as before,

$$Z(f) = \int_{-\infty}^{\infty} \left[\int_{-\infty}^{\infty} h(\tau)\, g(\tau + t)\, d\tau \right] e^{-j2\pi f t}\, dt$$

$$= G(f) \left[\int_{-\infty}^{\infty} h(\tau) \cos(2\pi f \tau)\, d\tau + j \int_{-\infty}^{\infty} h(\tau) \sin(2\pi f \tau)\, d\tau \right] \qquad (20)$$

The term in brackets is the complex conjugate of H(f) and is denoted by H*(f) in the final form of the theorem below.

$$\mathcal{F}\left[\int_{-\infty}^{\infty} h(\tau)\, g(\tau+t)\, d\tau\right] = G(f)\; H^{*}(f) \tag{21}$$

We will now continue on to the discrete Fourier transform starting with a continuous waveform h(t). The waveform is sampled or multiplied by a string of delta functions, s(t).

$$h(t)\, s(t) = \sum_{m=-\infty}^{\infty} h(m\Delta)\; I(t-m\Delta) \tag{22}$$

where capital delta is the sampling interval. The infinite sum is not realized and must be windowed, in this example, by $w(t)=1$ for $0 \leq t \leq (M-1)\Delta$ and zero elsewhere. So that now,

$$h(t)\, s(t)\, w(t) = \sum_{m=0}^{M-1} h(m\Delta)\; I(t-m\Delta) \tag{23}$$

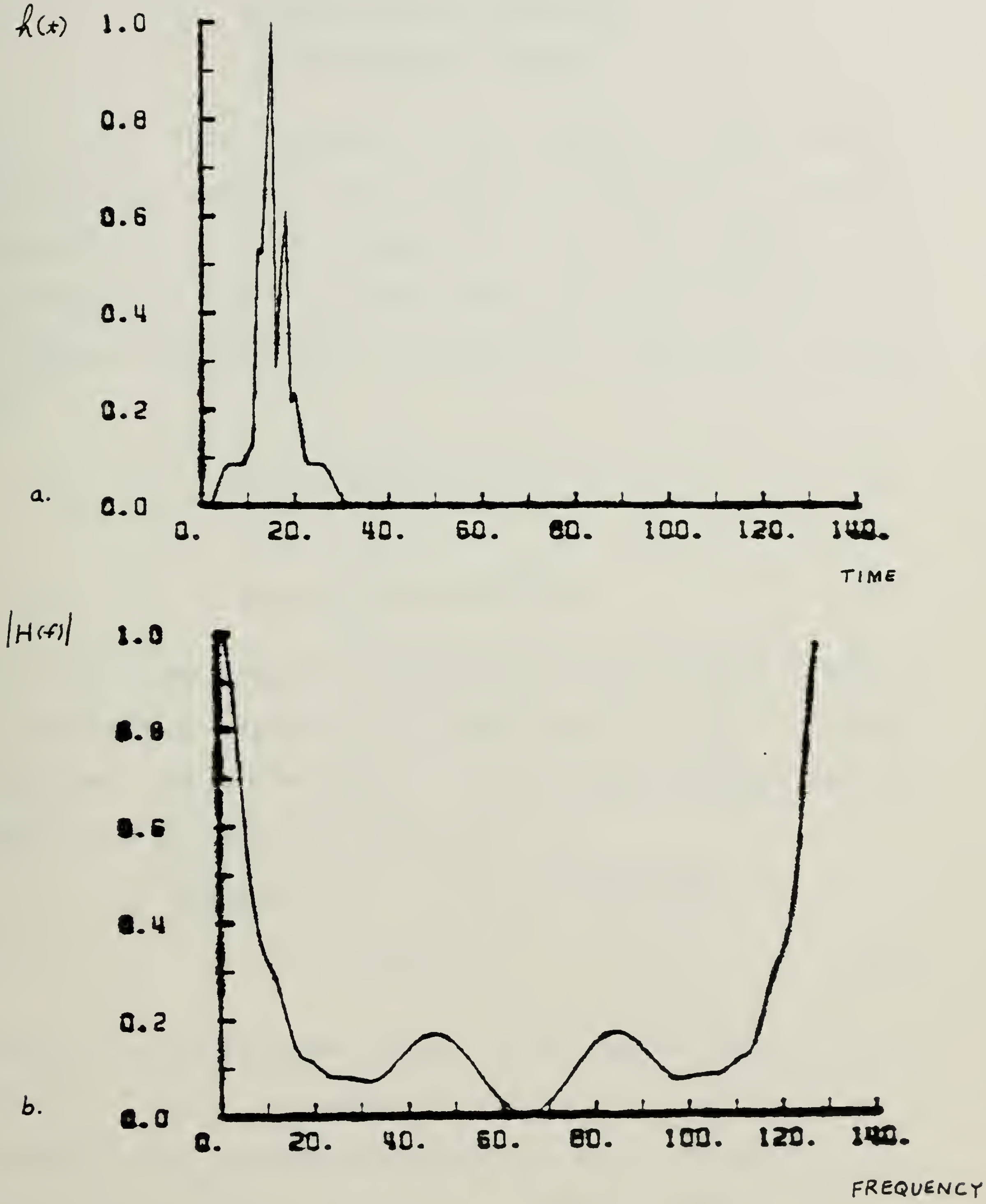
Recalling the convolution theorem, the multiplications in the time domain correspond to convolutions in the frequency domain with the following results. H(f) is convolved with the window functions´ spectrum and will have the apparent effect of introducing ripples because of the window´s significant sidelobes. The rippling may be minimized by choosing a window function with small sidelobes at the cost of other, perhaps more acceptable, spectral degradation.

H(f) is convolved with the sampling function and S(f)=I(f-n/Δ) has made the spectrum periodic with respect to the interval F=1/Δ. The spectrum coming from a real waveform is first symmetric about f=0 as shown before. Now because of its periodocity the spectrum is symmetric also about f=F/2 (the Nyquist or folding frequency). One final step remains. The Fourier spectrum is also taken at discrete points 1/T apart. The result in the time domain is the convolution of the sampled, windowed signal with I(t-nT), which is a periodic signal with T as its period.

The FFT algorithm used in the preprocessors developed in this thesis uses a Cooly-Tukey, base two algorithm [15]. This is documented where it is used in programs included in the Appendices. The algorithm uses N samples where N is two to an integer power. In the transformed domain, due to the symmetry shown above, there are N coefficients (only N/2 of which are unique as shown in figure 4). The original waveform must be band limited prior to sampling to minimize aliasing. The resulting frequency spectrum should approach zero at the folding frequency where up to half of the power can be aliasing noise.

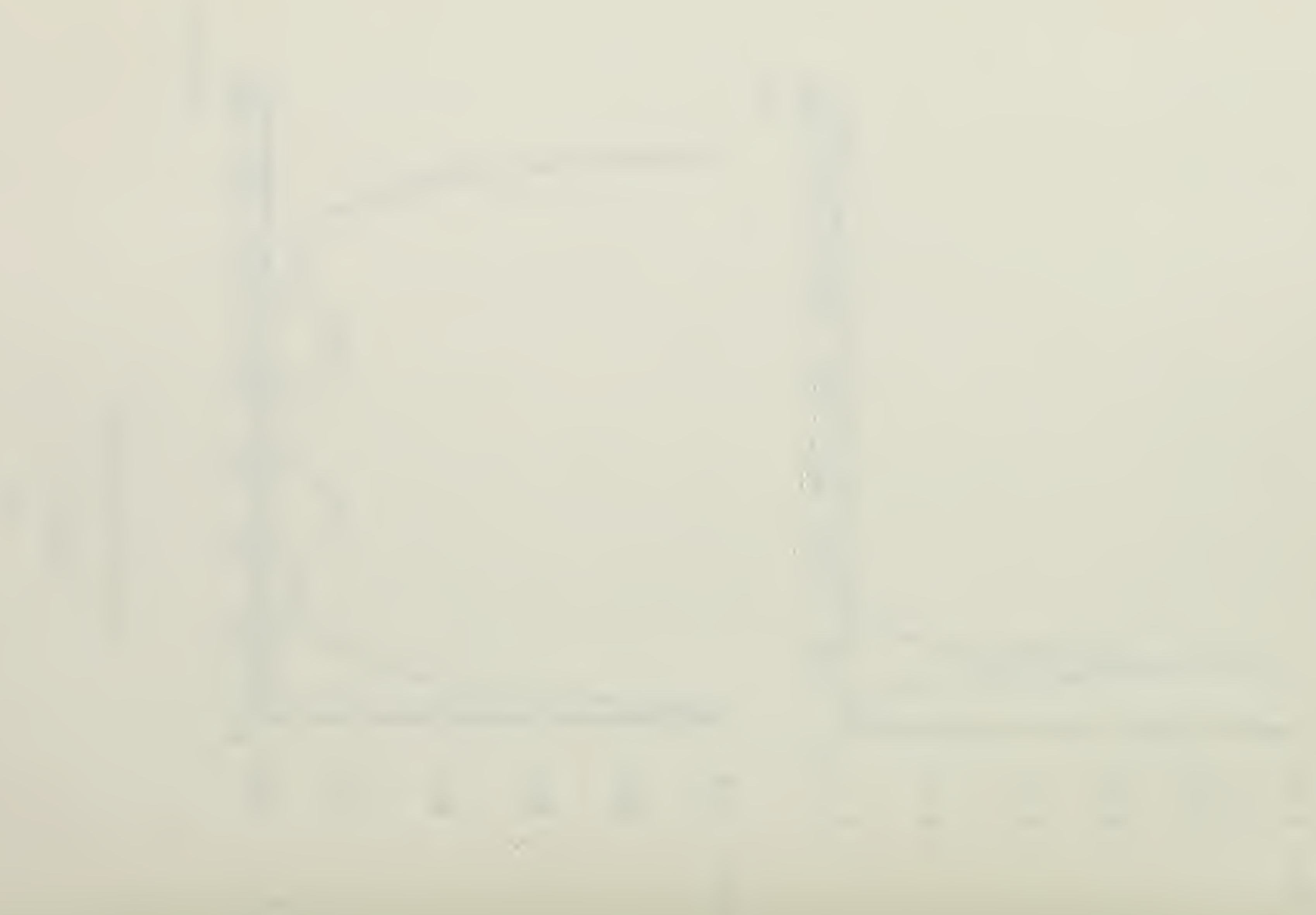
In the following two chapters, several different preprocessing algorithms are developed and their performance compared with canonic inputs. In Chapter IV the

problem of ship identification with range only radar is discussed briefly and one preprocessor is applied to several ship profiles. In the fifth and final chapter, conclusions are drawn from the work done here and follow-on efforts are recommended.



An FFT Spectrum

Figure 4



II. DIGITAL MELLIN TRANSFORM BY EXPONENTIAL WARPING

In the first chapter, the modulus of the Mellin transform was shown to be invariant to scaling. A detailed examination of the mechanics involved suggests an implementation that is widely used. The Mellin transform of a t-domain function h(t) is given in equations (7) and (9′).

$$H(s) = \int_0^\infty h(t)\, t^{s-1}\, dt \tag{7}$$

$$H(s) = \int_{-\infty}^{\infty} h(\Delta e^{x})\, e^{sx}\, dx \tag{9'}$$

Delta has been added, corresponding to the sample interval in the t-domain. Again it is noted that (9′) is a Fourier transform, where $s=-j2\pi m$. Solving the integral for the effect of a t-scaling by the factor k,

$$\int_0^\infty h(t/k)\, t^{s-1}\, dt = \int_{-\infty}^{\infty} h(e^{x-\ln k})\, e^{sx}\, dx$$
$$= e^{s\ln k}\, H(s) \tag{24}$$

Clearly, in the Fourier integral, the scaling factor k has become a shift for which the modulus of the transform is invariant. The exponential warp alone has transformed the scale factor into a shift. A prerequisite is that the t-domain signal has no shift. If there were a shift, it does not transform to a simple factor or shift in the

Mellin domain and so cannot subsequently be removed by taking the magnitude of a Fourier transform.

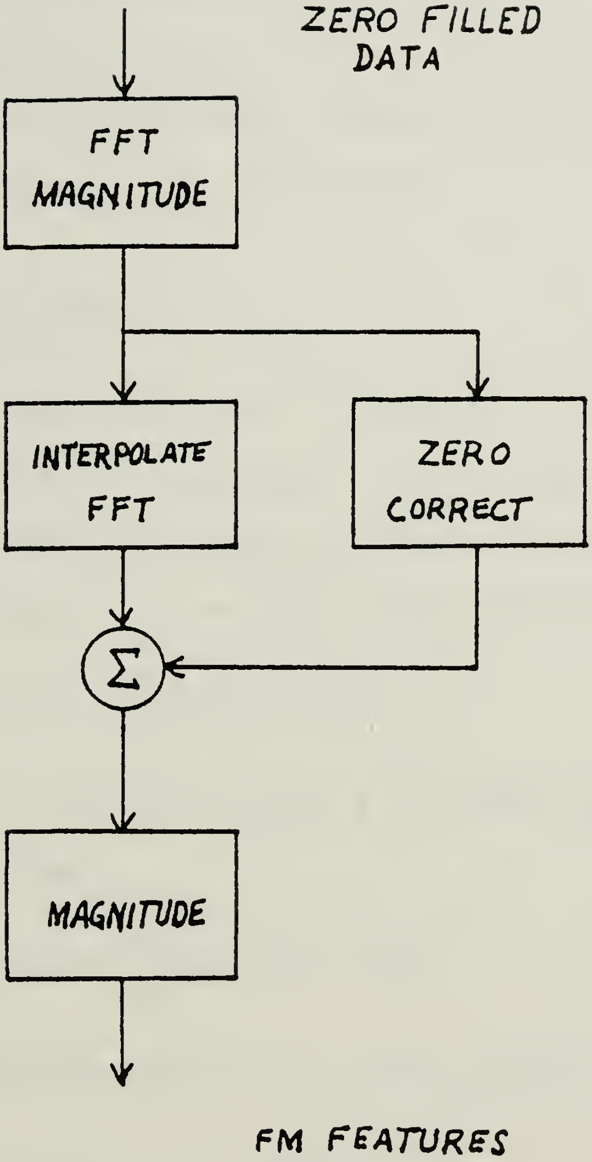
Implementation of a discrete Mellin transform is as difficult as it is with the Laplace transform [13]. Once transformed, characteristics in the new spectrum are difficult to relate to the original signal characteristics. One hypothesis relating the two domains is generated by comparison to the Fourier transform. The power spectrum associated with the Fourier transform can be used to detect periodicities in the physical function, since the wave numbers at which sharp peaks of the spectrum occur give the wavelength of such periodicities. By analogy, the positions of the peaks in the spectrum associated with the Mellin transform are said to give the magnification or compression which will produce features in the physical domain. Further, this stretching and compressing is identified as periodic in nature [21]. This seems unlikely because the Mellin is invariant to magnification/compression and does not behave well (a scale factor that is a function of t and k(t) as seen in (24). The Fourier spectrum models the original signal by a set of weighted sinusoids of varying frequencies, and therefore, naturally display periodicity and is invariant to shift. The Mellin is also a weighted

sum of sinusoids, but whose magnitudes are inversely weighted by t.

$$H(s) = \int_0^\infty h(t)\, \frac{e^{-j2\pi m \ln t}}{t}\, dt \tag{25}$$

Values for h(t) for 0 t 1 are far more important to the sum than those beyond that point. This characteristic contributes to the difficulty of realizing discrete Laplace and Mellin transforms and is a major topic covered in this chapter.

The numerical approximation of the Fourier-Mellin (FM) transformation by exponential warping is functionally diagrammed in figure 5. A FORTRAN program using the algorithm described in this chapter is included in Appendix A. Referring to figure 5, the input samples are from a pulse whose duration is finite and less than that of the sample window. For this case, no spectral distortion is experienced by filling zeros behind the sampled data. The only effect of the zero filling is to add spectral resolution in the frequency domain. Once through the first FFT block, and the magnitude taken, Ef is symmetric about zero and at the folding frequency. Thus, for N time samples and filled zeros, there will be N/2 unique spectral coefficients. This unique spectrum is interpolated, resampled as a warped signal and fed to the final FFT block. A correction is added, and the modulus is taken. The



Fourier-Mellin by Exponential Warping

Figure 5

resulting FM features are invariant to shifting and scaling in the time domain. The FFT block was covered in sufficient detail in the preceding chapter. Some effort will be spent in discussing the warping itself and the need for, and the development of, a zero point correction.

A. ALGORITHM DEVELOPMENT

This section uses its own notation to address the requisite exponential sampling. The series to be transformed is h(f). Its Mellin transform is H(m) where m is the Mellin frequency in $s=-j2\pi m$. The transform is,

$$H(m) = \int_0^\infty h(f) f^{s-1} df \tag{26}$$

Letting $f=F\exp[x]$ as before, where F is added corresponding to the sample interval

$$H(m) = \int_{-\infty}^\infty h(Fe^x) e^{sx} dx \tag{27}$$

The need is to evaluate M equally spaced samples in x, at

$$0, X, 2X, \ldots, (M-1)X \tag{28}$$

while the data consists of N equally spaced samples in f,

$$0, F, 2F, \ldots, (N-1)F \tag{29}$$

Assuming that F is a small enough interval to properly characterize h(f), it is sometimes advisable [9,10] to

choose the exponential sampling interval (X) such that the largest intersample spacing in h(Fexp[x]) is equal to kF where k=1. The other set of conditions used to uniquely specify the new samples are that f=F and x=0 will be the lowest limit for interpolation, while (N-1)F and (M-1)X are equated as the upper limit. The first requirement constrains the choice of M by

$$k = e^{(M-1)X} - e^{(M-2)X} \tag{30}$$

Meeting the second condition, the end points in each sampled series are equated yielding,

$$(N-1) = e^{(M-1)X} \tag{31}$$

Substituting (31) into (30) while applying the exponential series approximation gives,

$$e^{X} = 1 - \frac{k}{N-1} \doteq 1 - X + \frac{X^2}{2!} - \cdots$$

$$X \doteq \frac{k}{N-1} \tag{32}$$

One more substitution, (32) back into (31) sets up the desired result where M is now specified to exponentially sample from f=F to f=(N-1)F with kF being the largest interval between samples.

$$M \doteq \frac{N-1}{k} \ln (N-1) \tag{33}$$

As N gets larger, M explodes. If N=16 then M=41, if N=32 then M=106, if N=64 then M=261, and so on. This strict requirement can be compromised depending upon the application. The other extreme [18] is the criterion that requires the smallest interval between samples to equal the interval between uniformly spaced Nyquist sampling, with the intervals increasing exponentially thereafter. The specification permits analyzing frequencies approaching the largest which can be analyzed with uniform spacing. This greatly reduces the number of samples and decreases the required processing time, but is limited in application. Two factors mitigate the stringent requirements imposed by (33) where k=1. First, using N unique, uniform samples yields N/2 unique, uniform samples in the FFT domain. A related consideration is that the values of the spectral components approach zero at the folding frequency because the original signal has been band limited and some over sampling is normally recommended. Secondly, the inverse f weighting apparent in equation (25)

$$H(m) \doteq \int_0^{f_n} \frac{h(f)}{f} e^{-j 2\pi m \ln f} df \qquad (25')$$

attaches a decreased importance to h(f) near the folding frequency fn. These two effects combine to permit a much lower sampling rate once the modulus of the FFT has been taken. In spite of this, the stiffer rule (33) is used for

this work to generate the best FM domain possible. Whatever rule is used, once the exponential sample points have been computed for an FM preprocessor, they needn't be recomputed, but can be stored for rapid access during the interpolation. Some interpolation must be performed to approximate the spectral values of the new sample points. Third or even forth order Lagrange polynomials have been recommended and used for this purpose with apparent success [9,10,19]. The advantage of the Lagrange technique is that its notation is particularly compact, consisting of simple summations and repeated products that lend themselves to digital realization. Unfortunately, for the data sets used in this thesis, the third order algorithm was observed to behave poorly, adding a ripple in regions of rapidly changing slope. That this might be the case was suggested by the advice that Lagrange is very good near the central data point when the order of the polynomial is known to be the same as the order of the approximation, otherwise it is best left alone [20,21]. In its place, a second order polynomial was used to interpolate the warped samples with results that were nearly indistinguishable from the actual waveform, as seen later in figure 8.

3. ZERO POINT CORRECTION

Another problem becomes apparent when the Mellin transform of h(f) is recalled,

$$H(s) = \int_0^\infty \frac{h(f)}{f} f^s df = \int_{-\infty}^{\infty} h(e^x) e^{sx} dx \qquad (34)$$

where $s=-j2\pi m$. The exponentially sampled waveform described above is applied to an FFT block. As f approaches the folding frequency, h(f) tends to zero. Unfortunately, as f approaches zero, the value of h(f) is not zero. In fact it is frequently rather high. To make matters even more interesting, the left integral in equation (34) clearly shows that the closer to zero f gets, the more important h(f) becomes to the integral. Several solutions to this problem are considered below.

One practical, simple approach is to set the DC (ie, f=0) term of the FFT to zero. The effect is nothing more than removing a DC level back in the signal domain, but Mellin transformation into the FM domain leaves the spectrum dependent upon the scale factor k [22]. Setting the f=0 coefficient to zero corresponds to setting h(f) to zero for 0 f<1, where the unity upper limit is chosen

without loss of generality. The resulting transform of a scaled signal domain h(f/k) is

$$\int_1^\infty h(f/k) f^{s-1} df = k^s \int_k^\infty h(f) f^{s-1} df \tag{35}$$

which is obviously dependent upon k. The closer k is to unity, the smaller the effect. By increasing the f spectral resolution, the error can be reduced. The error may be insignificant for many applications [5-8], but the technique should be used with care.

The first solution has highlighted the need for a zero point correction. Another common solution [10-11] is developed by breaking the integral up as before. Again using unity as the upper limit of the left integral while remaining general in application,

$$\int_0^1 h(f) f^{s-1} df + \int_1^\infty h(f) f^{s-1} df$$

$$\doteq \frac{h(0)}{s}\left[1^s - 0^s\right] + \int_1^\infty h(f) f^{s-1} df \tag{36}$$

Two assumptions are made to get the correction term. First, that h(f) remains a constant h(0) over the interval of f between zero and one. Second, and not as easily accepted,

$$0^s = e^{-j\omega \ln 0} = e^{j\infty} = 0 \tag{37}$$

Equation (37) pretty well shows why this assumption is suspect, but playing along for the momment, the question is

reserved for a later detailed look. Accepting the assumption, the correction factor becomes,

$$Z(s) = \frac{h(0)}{s} = -j\,\frac{h(0)}{2\pi m} \tag{38}$$

Because H(s) is a complex function, Z(s) must be applied (added to the imaginary part of the succeeding Fourier transform) before the magnitude is take to remove scaling dependence. This correction is specifically derived for use with a continuous Fourier transform such as the optical Fourier in imaging systems with the added stipulation that h(f) be nearly constant over the range $0<f<k$ where k is the largest scale factor expected. If an FFT is employed to make this final transform, another correction should be applied as shown by Zwicke and Kiss [11] below. This correction factor differs from the first due to an invariant property of the FFT. The FFT of two unit step functions that vary only in scale are balanced.

$$\left| Z_m(\omega) + \sum_{i=0}^{m} e^{-j2\pi k i/M} \right| = \left| Z_p(\omega) + \sum_{i=0}^{p} e^{-j2\pi k i/M} \right| \tag{39}$$

where m and p are arbitrary integers greater the zero and less that M. Successive FFT coefficients are summed, and the average value of the contributing terms taken resulting in,

$$C = \frac{1}{2} - \frac{j}{2}\cot(\pi k/M) \tag{40}$$

This is then multiplied by h(0) to arrive at the FFT Mellin correction factor.

$$Z_{FFT}(k) = \frac{h(0)}{2}\left(1 - j\cot(\pi k/M\right) \qquad (41)$$

When k/M is small the imaginary term dominates and the correction factor approaches that used in the continuous case (38). Most of the work done for the thesis on the exponential algorithm was done using the inappropriate zero point correction (38). Since its discovery was coincident with that of more powerful methods discussed in Chapter III, little data was taken using (41).

To bound the error involed, an acceptable h(t) is defined, windowed and transformed using the Mellin integral. The window limit is then allowed to grow unbounded and the resulting expressions are interpreted as the error. It's been assumed that for 0<t<1, h(t)=h(0). Since the error resulting from the assumption in equation (37) arises from this interval alone, t>1 is ignored for the time being. Warping the signal as before h(x)=h(0) for all x<0 and h(x)=0 for all t>0. Evaluating the integral to a finite window width (T),

$$h(0)\int_{-T}^{0} e^{-j\omega x}\,dx = j\,\frac{h(0)}{\omega}\left(1 - e^{-j\omega T}\right)$$

$$= \left[T\,h(0)\left(\frac{\sin(\omega T/2)}{(\omega T/2)}\right)\right] e^{-j\omega T/2} \qquad (42)$$

The term in brackets is the magnitude of the contribution from the region of integration, $-T<x<0$. Note that it involves a sin()/() term. The effect is to add a peak of (T)h(0) at the origin. The size of the peak depends directly on the window border T. Letting T approach infinity raises the spike at $\omega=0$, with lesser peaks at $\omega=(2i+1)\pi/T$, where i is an integer. Each of the subpeaks has a magnitude of $2h(0)T/((2i+1)\pi)$. Substituting ω into the second relation yields the envelope (in brackets) and phase as T tends to infinity.

$$Z_b(\omega) = \lim_{T\to\infty} \left[\frac{2h(0)}{\omega}\right] e^{-j\omega T/2} \qquad (43)$$

The error bound in brackets, does not depend on the sample rate, or the size of the window. Any approximation approaching zero will have the same bound. Although the magnitude of the error is in a convenient form, the phase is indeterminate. For the correction to be applied, the complex addition must occur prior to the modulus being taken. This cannot be done, leaving the error uncorrected, but is bounded by $2h(0)/\omega$ for continuous and aperiodic discrete Fourier transforms. For the FFT, equation (43) does not bound the error. The sum of 1/i does not converge as i tends to infinity. At any point on the FFT this sum is present due to the apparent folding. The error itself is not unbounded because phase differences in the sum of the

errors at any point may result in the envelopes adding destructively, reducing the actual error. Adding the envelopes is an unrealistic, worse case approach. The FFT correction (41), can not be compared to (43) for these reasons. The other error correction, using the assumption in equation (37), can be compared. The first, setting $h(0)$ to zero, or just plain ignoring the $0<t<1$ interval have the error function bound in equation (43). Although only differring by a factor of two in magnitude, the constant phase is arbitrary, and equivalent to setting $T=0$; that is, assuming $h(0)=0$ over $0<t<1$. This was the very problem the correction was developed to remedy, but is without effect. Equation (41), the correction for the FFT is not completely accepted by this author, albeit no real empirical evidence have served to verify or dispute the claim. Suspicions are raised on two aspects. The indeterminate phase of the error (43) in the continuous case arises naturally from approaching the $t=0$ limit with the Mellin integral. This quality is conspicuous by its absence in the FFT error correction. The FFT error correction is computed by summing the FFT coefficients in the complex plane. The average position of the resulting polynomial is the error correction term. However, the FFT coefficients of a finite duration signal are the values of the signal, evaluated at M evenly spaced points about the unit circle [3,23]. If

the sequence is a constant, the average value is the origin of the complex plane.

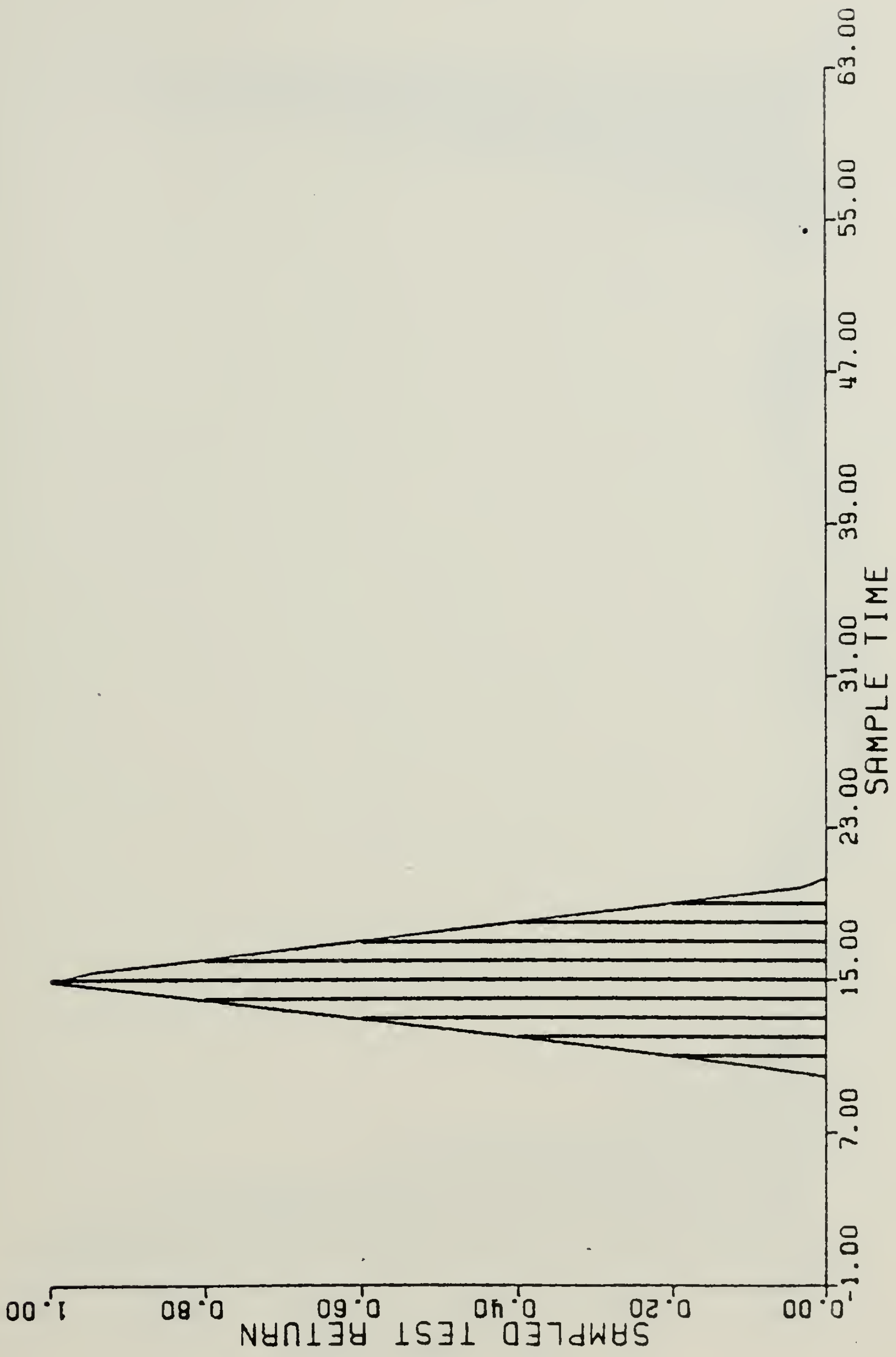
The zero point corrections for the Mellin transform are unbounded at $\omega=0$. More time could have been spent determining the best applied correction for the specific case at hand, but direct methods are developed in the next chapter that obviate the need to employ the correction at all.

C. TESTS AND RESULTS

It is worth admitting at this point that the results using the exponentially warped algorithm to achieve a discrete Mellin transform have not been good. More recently developed techniques in Chapter III greatly surpass the results reported in this section. Although much of the theory used to improve performance in the following chapters could have been used here, this was a preliminary attempt that was later abandoned. The FM processor described in the previous section was built using FORTRAN. Appendix A is the documented program. This section will review the processing with actual plots of the signatures at different stages, discuss the required tests, introduce the testing approach, and finally intepret the results. The functional block diagram of the preprocessor, figure 5, is

near the beginning of this chapter and could be profitably reviewed. Figures 6 through 9 represent the step by step view of the signal processing, where the signal, a test shape, is shown in figure 6. The waveform, appears as an envelope, and is drawn with vertical lines indicating the sampled series. Figure 7 is a picture of the FFT, in its sampled version showing its symmetries. Figure 8 is a very close approximation to the continuous periodic transform achieved by filling zeros to obtain the requisite spectral resolution. A "+" on the transform plot indicates an exponential sample point interpolated from the sixteen unique points in figure 7. The warped samples are sent through the FFT block once more with the result shown as figure 9. Heavy spectral coloring by the $h(0)/\omega$ correction factor is evident. Only the first half of the spectrum, from zero to the folding frequency, is valid.

Complete scale invariance was never realized although its effect was greatly reduced. All the testing done on the exponential algorithm was an attempt at achieving and verifying shift and scale invariance. Much effort was spent structuring the tests to avoid the effects of processing noise so the actual algorithmic characteristics could be determined. Along the way, requirements arose to select a suitable interpolation polynomial, an optimal zero point correction and other system improvements. In all cases, the

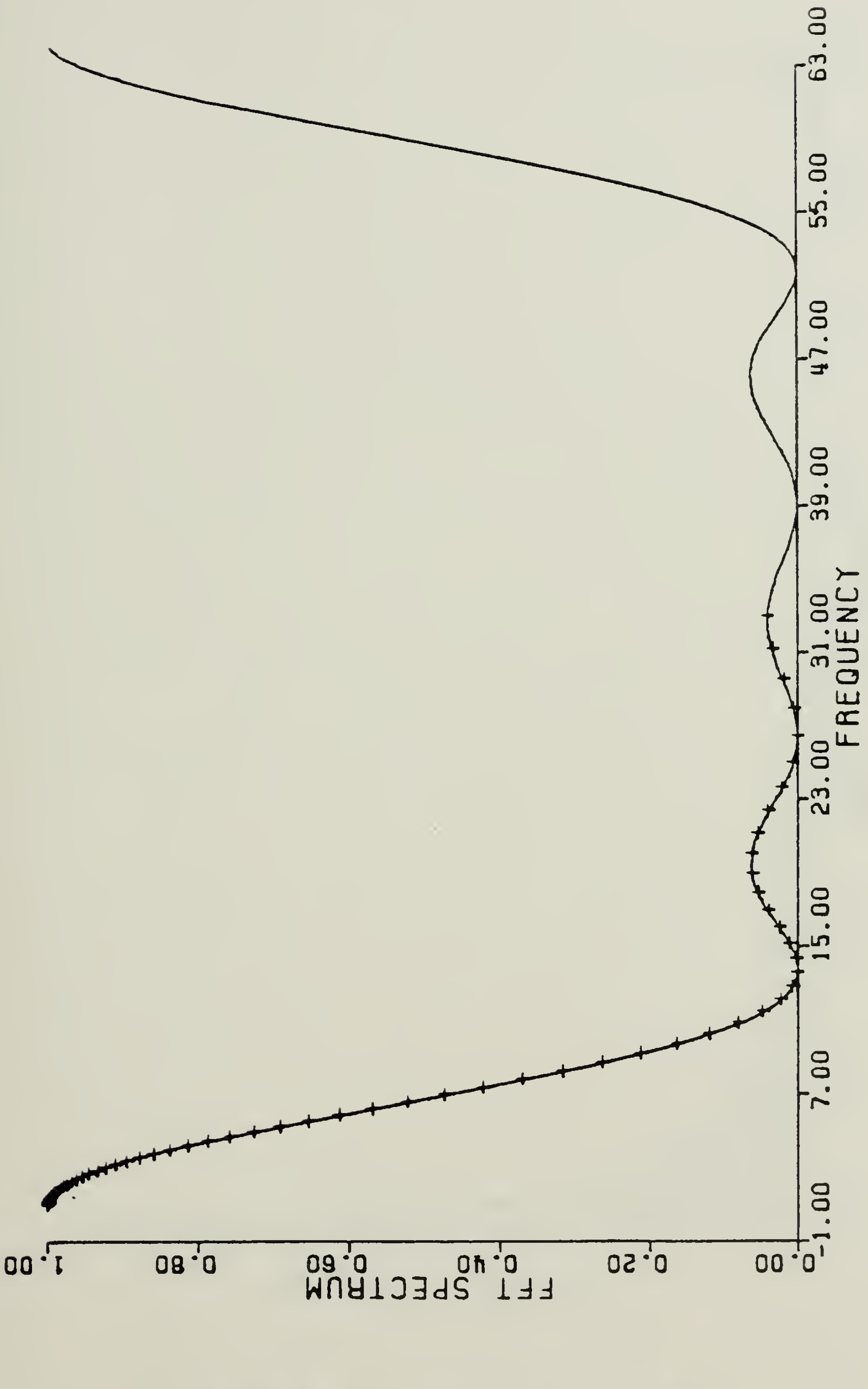


Test Shapes

Figure 6

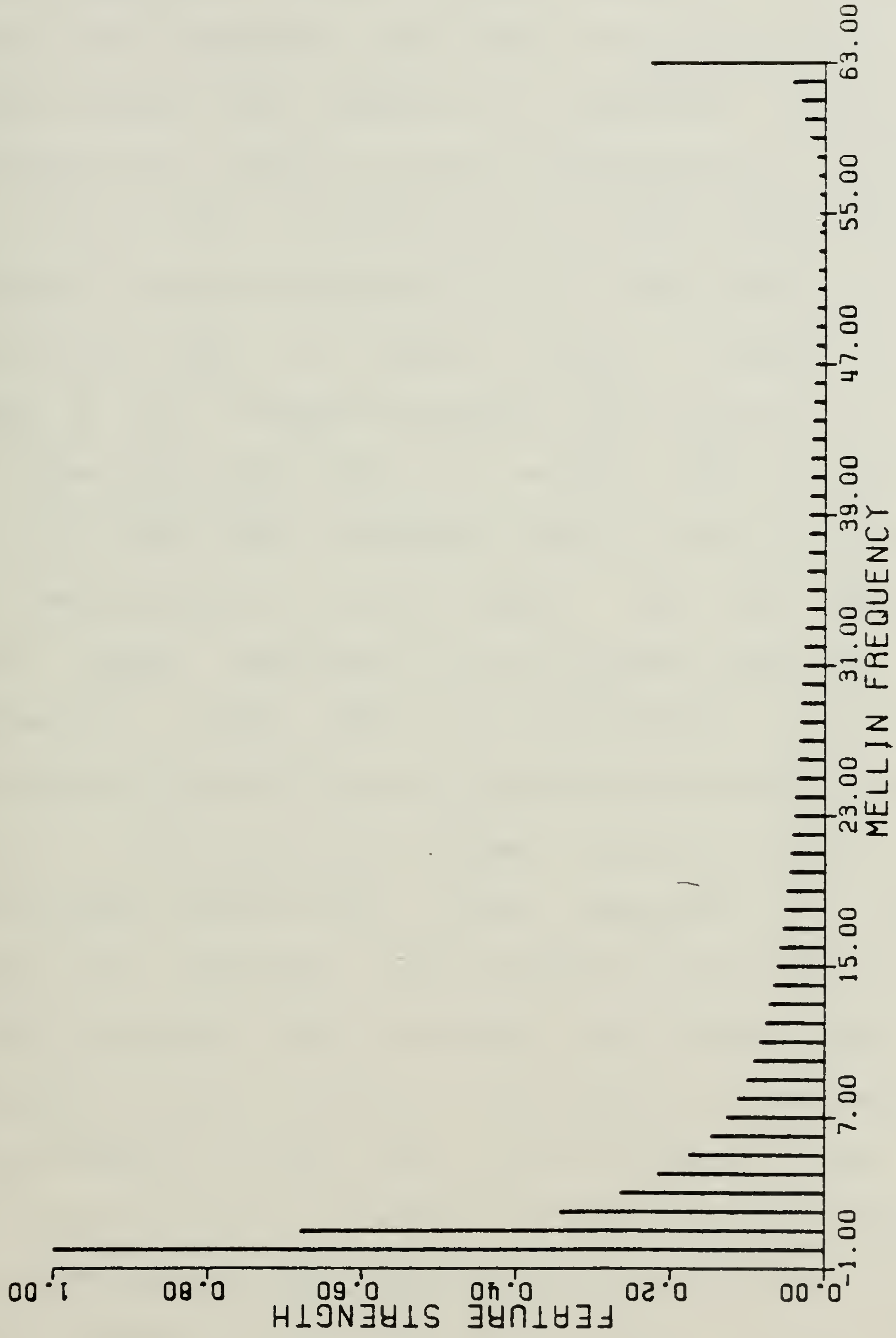
Test FFT

Figure 7



Test Exponential Sampling

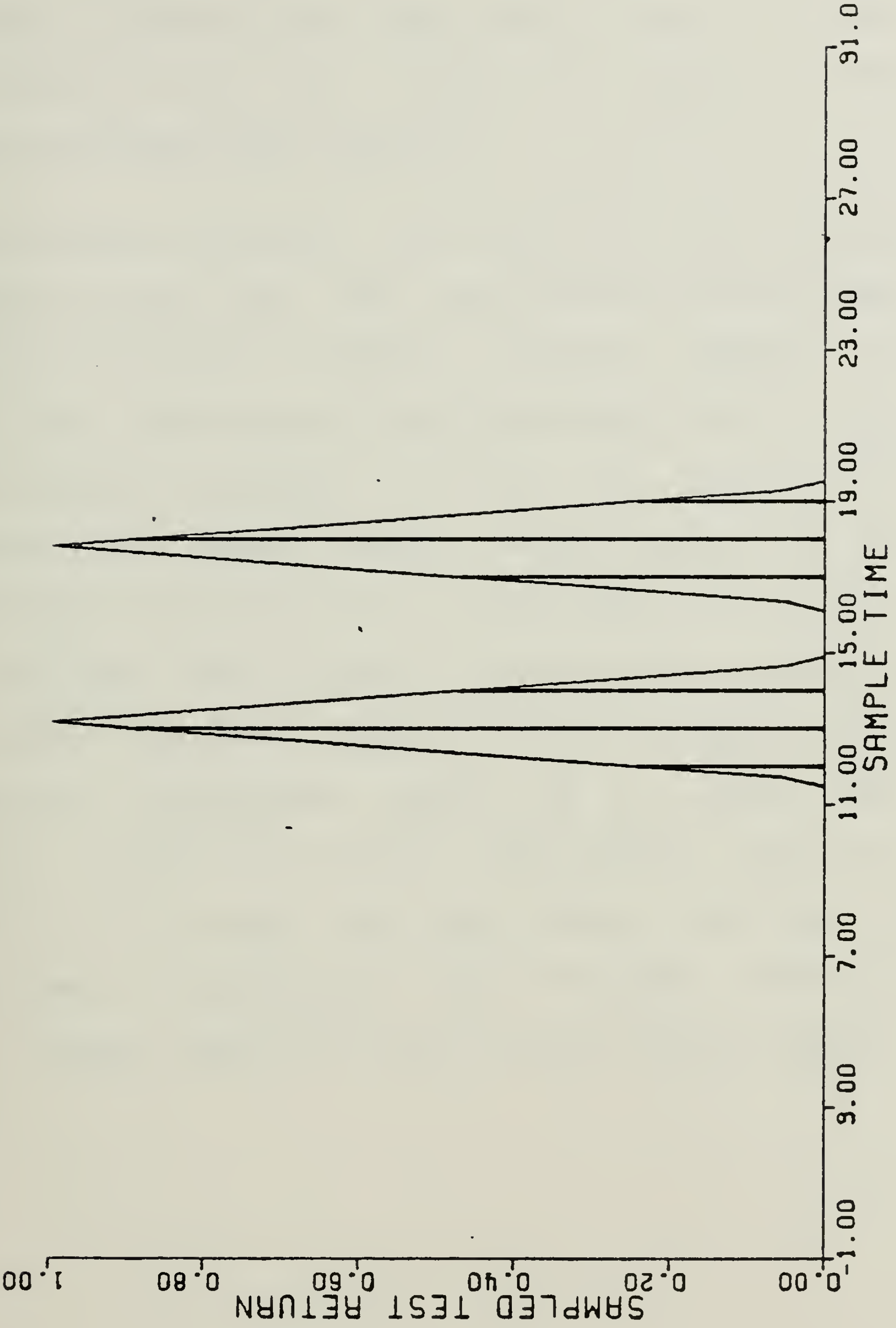
Figure 8



Test Fourier-Mellin Domain

Figure 9

test procedure was to first select canonic test shapes. Squares and triangles were most frequently used, being shifted, scaled and combined to determine preprocessor characteristics. Scaling was most frequently by a factor of two or less. This corresponds to an aspect angle change of 60 degrees from the unscaled case. For many tests, care was specifically taken to exactly scale a sampled series instead of the waveform envelope. The variation in input signal when this isn't done is evident when considering figure 10. For the envelope shown, the sampled series cannot reconstruct the same signal, and may result in feature space variation. Two feature qualities were monitored in each test to determine preprocessor performance; insensitivity to shifting and scaling, and the ability to differentiate between canonic classes. These qualities were measured by visual comparison of the FM features, by computing the correlation coeffiecients and the mean squared error between the feature sets of differently scaled similar test shapes, and by computing the distribution of the error over the feature space. The latter test was an attempt to locate feature regions of class commonality, and regions of distinction between canonic classes. This approach allowed macroscopic and microscopic examination under changes of scale, shift and shape. The clustering and separation qualities are data



Sampling Variation

Figure 10

dependent. Since no ship radar video data was used, all observations were with respect to canonic classes. No strong groupings were detected.

As alluded to earlier, the results for the exponential algorithm were less than satisfactory. Test shapes were carefully designed to minimize sampling effects, and to ensure low side lobes in the frequency domain. Many tests were conducted using each of the zero point correction methods with varied shapes, sample rates, and spectral resolutions. Classification on the basis of signal shape was very poor. The strongest correlation was between shapes of common duration. Table 1 shows a typical result of comparing a rectangular shape and a raised ramp. The scaling in each case was by 2 (60 degrees). Equation (38), $Z=h(0)/\omega$ was used, but the others offerred little improvement. Consistently, the strongest similarity was shown between shapes that had the same sample length, vice shape.

TABLE 1

Canonic Shape Fourier - Mellin
Feature Comparisons

a. Peak Correlation Values

	RECT	RECT/2	RAMP	RAMP/2
RECT	1.00	0.77	0.98	0.76
RECT/2	-	1.00	0.87	1.00
RAMP	-	-	1.00	0.86
RAMP/2	-	-	-	1.00

b. Squared difference between features.

	RECT	RECT/2	RAMP	RAMP/2
RECT	.000	.032	.019	.032
RECT/2	-	.000	.032	.019
RAMP	-	-	.000	.032
RAMP/2	-	-	-	.000

III. DIRECT MELLIN TRANSFORMS

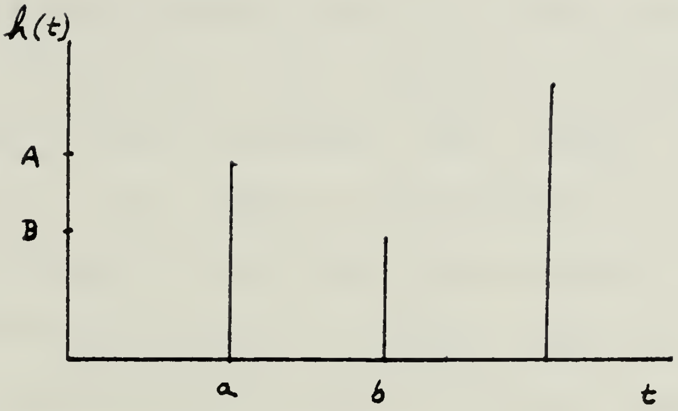
The last chapter developed a method of obtaining the Mellin transform by exponentially warping the signal prior to using an FFT block. This technique is referred to as the fast Mellin transform (FMT). Although the promise of scale invariant features is attractive, some of the problems encountered that make the FMT unattractive are reviewed here. The required sample rate varies with respect to the data, making general applications difficult. The tendency is to use more samples than required, which quickly becomes costly in an exponential sampling scheme. The need to exponentially warp (to interpolate) a set of new samples, is expensive in time required. For true scale invariance, a correction factor is required but because of the integral's unbounded nature at zero this correction factor is indeterminant. Several correction methods have been employed, but they depend on unspecified data characteristics. These effects combine to make the actual performance of the algorithm poor. Although scaling effects are mitigated, they remain an artifact, which is disturbing to classification attempts. This chapter outlines the effort to remove these limitations. Some useful Mellin properties are developed, and then applied to establish

several Direct Mellin Transforms (DMTs) which were built, and their performance compared.

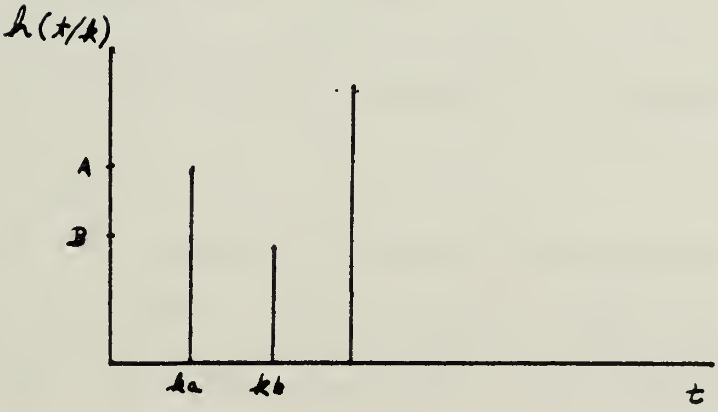
A. SOME USEFUL PROPERTIES

Some general observations are made here about the Mellin transforms, and are followed by some specific relationships which were derived and applied. A property of the Mellin s-domain is that it is unaffected by scaling changes in the original x-domain. Figure 11a is a test shape in the x-domain. Two features are identified according to their amplitudes A and B, at x=a and x=b respectively. The ratio of a/b equals c. To be simply scaled by k, h(x/k) must remain the same in all aspects except that the distance between features has been changed according to the scale factor k. Figure 11b shows the scaled domain. Features A and B are again identified at their scaled positions ka and kb. The signal property maintained by scaling is relative positional integrity, that is, ka/kb equals c as before. The positional integrity of the features, the ratio of their distance from the origin, to that of another's is unchanged. Restated, an operation O[h(x)] in the x-domain, will leave the s-domain modulus invariant to simple scaling by k if

$$O[h(x/k)] = f(x/k) \qquad (44)$$



a. UNSCALED



b. SCALED

Pure Scaling

Figure 11

Note that the entire domain is scaled, so no unscaled shift in the domain can be permitted as already discussed. In the Mellin s-domain, any simple scaling in the x-domain results in a phase distortion (the modulus is invariant to k). Manipulations in the s-domain will leave that domain invariant to k, as long as the modulus is modified by a multiplicative factor of constant phase. That is, if $|G(s)|$ is an arbitrary function of s (except that it does not depend on k) and $|M[h(x/k)]|=|H(s)|$ is also invariant to k, then their product is invariant to k and the x-domain remains simply scaled. For instance, in the x-domain the operator $O[h(x)]=x(h(x))$, does not meet condition (44).

$$x h(x/k) \neq f(x/k) \tag{45}$$

So the Mellin transform's modulus of $xh(x/k)$ cannot be invariant to k.

In Chapter II, a error was apparent because h(x) was not equal to zero for x=0. If h(x) could be modified with an operation that met the condition of equation (44) and

always produced a series that was zero at x=0 a general approach can be developed. Consider two operators,

$$O[h(x/k)] = \frac{d}{dx}(x\,h(x/k)) = f(x/k) \tag{46}$$

$$O[h(x/k)] = x\frac{d}{dx}(h(x/k)) = f(x/k) \tag{47}$$

Equation (46) will produce an acceptable f(x) as df/dx=0 at x=0. Equation (47) will always produce the required condition. To see the frequency domain equivalents, we must assume that the Mellin integral exists. Integrating by parts,

$$\left| \int_0^\infty h(x/k)\, x^{s-1}\,dx \right| = \left| H(s) \right|$$

$$\left| \int_0^\infty x^s \frac{dh(x/k)}{dx}\,dx \right| = \left| \Big[x^s h(x/k) \Big]_0^\infty - s \int_0^\infty h(x/k)\, x^{s-1}\,dx \right|$$

$$= \left| -s \int_0^\infty h(x/k)\, x^{s-1}\,dx \right| = \left| s\,H(s) \right| \tag{48}$$

The result in the frequency domain is consistant with the conditions stated above. The limit of h(x) as x goes to zero or to infinity must be zero if equation (48) is to be

true. Similarly, under the same conditions, it can be shown that

$$\left| \mathcal{M}\left[\frac{d}{dx}\left(x\, h(x/k)\right) \right] \right| = \left| (s+1)\, H(s) \right| \tag{49}$$

Equations (48) and (49) clearly do not apply where h(0) is not equal to zero. However, by using the x(d(h(x))/dx) modifying operation of (47), a series can always be zero at x=0. The function h(x) is further constrained by the fact that it must fall off faster than 1/x. This assumption must be valid and the modifying operator must be applied in the x-domain for the Mellin integral to exist in general. A Mellin transform of a function, after having a modifier applied, will be called a modified Mellin transform.

$$H_a(s) = \int_0^\infty \frac{d\, h(x)}{dx}\, x^s\, dx \tag{50}$$

The integral is close to a form which is realizable, except for the upper limit. For a finite sampled series, h(n) will be assumed zero outside of the interval $0 \le n < N$. This truncation effects the transform of an otherwise infinite series. In this application the Mellin is applied to an FFT frequency spectrum. The truncation in the frequency domain is due to band limiting the signal prior to sampling. Scaling in the time domain will not result in simple scaling, but will add a dependence on the scale factor k that can not be removed by the transform. An approach by

Prost and Goutte is used to predict the size of the error [24,25]. First a suitable function will be selected and a relative error of truncation (RET) determined and applied to two scalings. The relative difference of the feature space is found and identified as the error. Remembering that this is applied to a frequency spectrum, dh(f)/df is approximated by a function of the form.

$$\frac{dh(f)}{df} \doteq f e^{-f} \tag{51}$$

The modified Mellin transform of (51) over a finite frequency range would be approximately,

$$H_a^{*}(s,F) = \int_{1/F}^{F} f e^{-f} f^{s} df \tag{52}$$

The lower limit has been set in a manner to be consistant with Plancherel´s theorem. A convenient worse case assumption is that the lower limit is essentially zero, but this depends, in general, on F and the data itself. Ha*(s,f) converges toward Ha(s), equation (50), in the mean square as F tends to infinity. The mean square error will not provide an expression for the error that can be used to correct for it but is useful to measure the effect of the truncation in more general terms. A relative error of truncation (RET) can be computed if the error is assumed to be distributed evenly over the range from which Ha*(s,f) is

computed. By employing Plancherel's theorem, the mean squared values are,

$$e_\infty^2 = \int_0^\infty \left| f e^{-f} f^s \right|^2 df = \int_0^\infty f^2 e^{-2f} df \tag{53}$$

$$e_F^2 \doteq \int_0^F f^2 e^{-2f} dF \tag{54}$$

The RET is defined and solved as,

$$RET = \left(\frac{e_\infty^2 - e_F^2}{e_\infty^2} \right)^{\frac{1}{2}} = e^{-F} \left(2F^2 + F + 1 \right)^{\frac{1}{2}} \tag{55}$$

But the limit F depends not only on the pass band F, but on the scaling in the f-domain. A relative error (RE) between a truncated spectrum and a scaled and truncated signal would be more complex, but worth the effort.

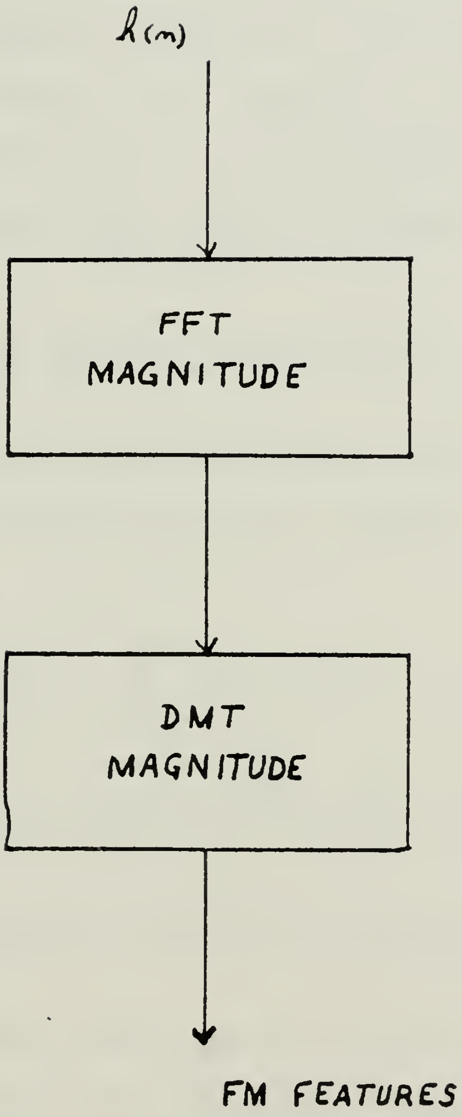
$$RE = \left(\frac{e^{-2F}(2F^2 + F + 1) - (1/k)\, e^{-2kF}(2k^2F^2 + kF + 1)}{1 - e^{-2F}\,(2F^2 + F + 1)} \right) \tag{56}$$

where 0<k<1. Two observations should be made here. First, the relative error of truncation (55) and the relative error or difference between two truncated spectrums scaled differently (56) both depend on F. F depends on the cut off frequency of the low pass filter prior to any sampling. It is often chosen to minimize aliasing depending upon the

limitations of the sampling circuit. Second, RE depends on k as well. The importance of scaling differences to this data type can be readily seen. If equations (55) and (56) are valid, and if the range of k can be bounded (a design specification) F can be chosen to realize a stated RE. Or if F is fixed and k bounded, the RE may be determined for evaluation. If the data type is not appropriate, a more representative function may be determined and used in place of equation (51) to attain better expressions for RET and RE.

B. ALGORITHM DEVELOPMENT

Part A above provided some background for making the Direct Mellin Transforms (DMTs) in this section. In this presentation the justification is given with the application first. However, chronologically the neat package presented above was preceded by extensive evaluation of empirical results, not vise versa. Appendix B documents the FORTRAN implementation of all the algorithms developed in parts 1 and 2 below. Figure 12 is a functional block diagram of a generalized FM preprocessor using a DMT to show the simplicity with which it can be applied, as opposed to the FMT covered in the second chapter, figure 5.



DMT Preprocessor

Figure 12

1. First Difference Approximations

Although developed through a different rationale, this first algorithm was developed by Zwicke and Kiss [11]. Starting with a sampled series hi, the series is operated on by the modifier defined in equation (51), using the first backward difference to approximate the derivative with respect to x. Unit step size is assumed.

$$x \frac{d h(x)}{d x} \doteq n \left(h_n - h_{n-1} \right) = n \nabla h_n \tag{57}$$

Taking the trapezoidal rule to evaluate the modified Mellin integral (52) while recalling that $h(0)=0$, and $h(N)$ is assumed zero,

$$H_{a1}(m) = \sum_{n=1}^{N-1} \nabla h_n \, n^s \tag{58}$$

where $s=-j2\pi m/M$. The complex coefficients are

$$n^s = \cos 2\pi (m/M) \ln n - j \sin 2\pi (m/M) \ln(n) \tag{59}$$

They can be calculated off line and stored to produce just the desired characteristics. The factor $(2\pi m/M)$ could be any number that produces an interesting feature. Undesired features need not be computed (what in general was a N by M matrix, where M is the number of Mellin transform coefficients and N the number of x sample points). If a relatively small number of features is required, perhaps

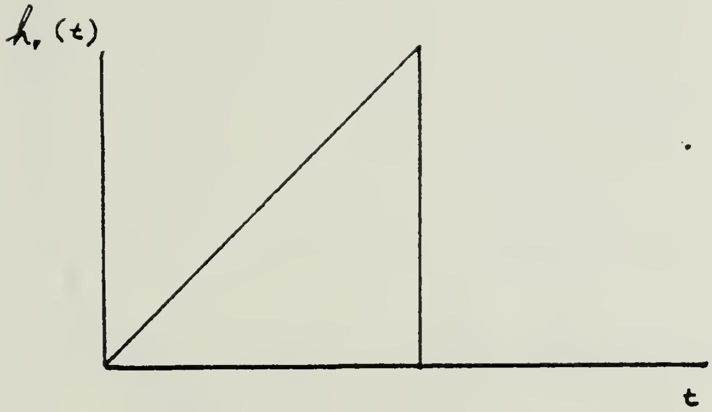
the processing will be manageable. Notice that no zero point correction is required. Only data changes contribute to the transform. These observations are valid for any of the modified Direct Mellin Transforms developed in this section.

By using a central difference instead of the backward difference, a similiar result is obtained.

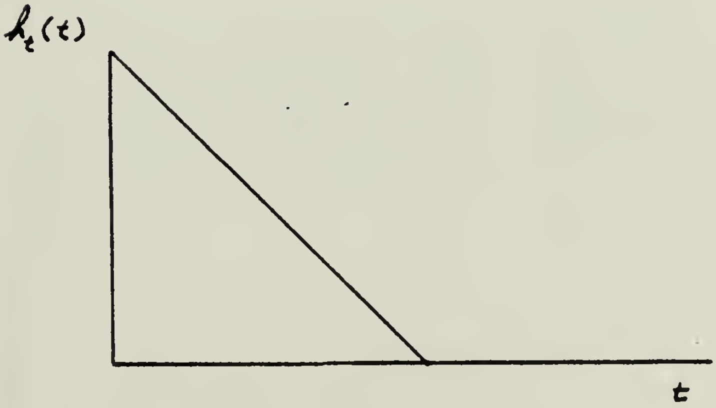
$$H_{a1}(m) = \sum_{n=1}^{N-1} (h_{n+1} - h_{n-1})\, n^{s}/2 \qquad (60)$$

Other numerical integrations may be used with improved results, and other methods can be used to increase the order of the approximation.

To test the algorithms, a ramp and inverse ramp were used in a scaled and unscaled mode. The ramps and their scalings are shown in figure 13. Figure 14 is the analytic results of both waveforms plotted with the transform found using equation (60). This is a dramatic improvement over anything used with the methods discussed in the previous chapter. The noise of the signal appears to be diverging as frequency grows, but over the range plotted, the appearance is that of an algorithm trying to do well.



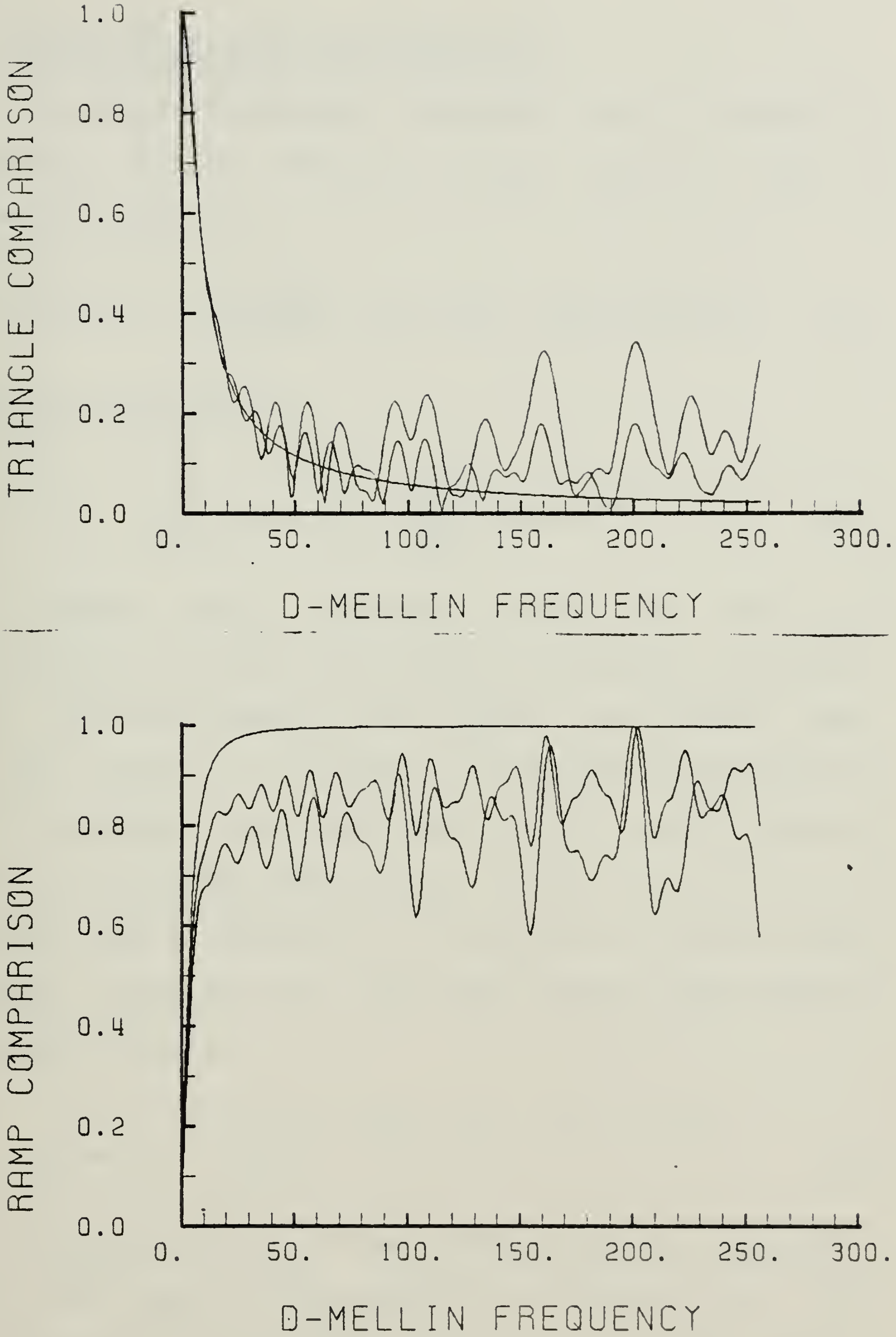
a. RAMP



b. TRIANGLE

Test Shapes

Figure 13



First Difference Results

Figure 14

2. Second Difference Approximations

A second difference algorithm can be achieved by proceeding as before. The pure scaling operator used to prepare the signal is,

$$O[h(x)] = x^2 \frac{d^2 h(x)}{dx^2} \doteq n^2(h_{n+1} - 2h_n + 2h_{n-1}) \qquad (61)$$

And the new algorithm is,

$$H_{a2}(s) \doteq \frac{1}{s+1} \sum_{n=1}^{N-1} \Delta^2 h_n n^{s+1} \qquad (62)$$

Other second order operators have been used, and partitioned into forward and backward difference variations to (61), but this appears to be a basic and useful form. The color 1/(s+1) is present to approximate the modified Mellin of equation (60). The term (s+1) is valid assuming that dh(x)/dx is exclusively upper bounded by ln(x)/x as it approaches zero or infinity. For comparison, another second difference algorithm was developed based on the modifier (x(d/dx)(x(d/dx)h(x).

$$H_{a2}(s) \doteq \frac{1}{s} \sum_{n=1}^{N-1} (n\Delta^2 h_n + (h_{n+1} - h_{n-1}))\, n^s \qquad (63)$$

This is roughly the sum of the methods defined in equations (60) and (62) above. The assumption for deriving the color 1/s is even less restrictive than before, but the algorithm performs poorly compared to (62). These transforms depend

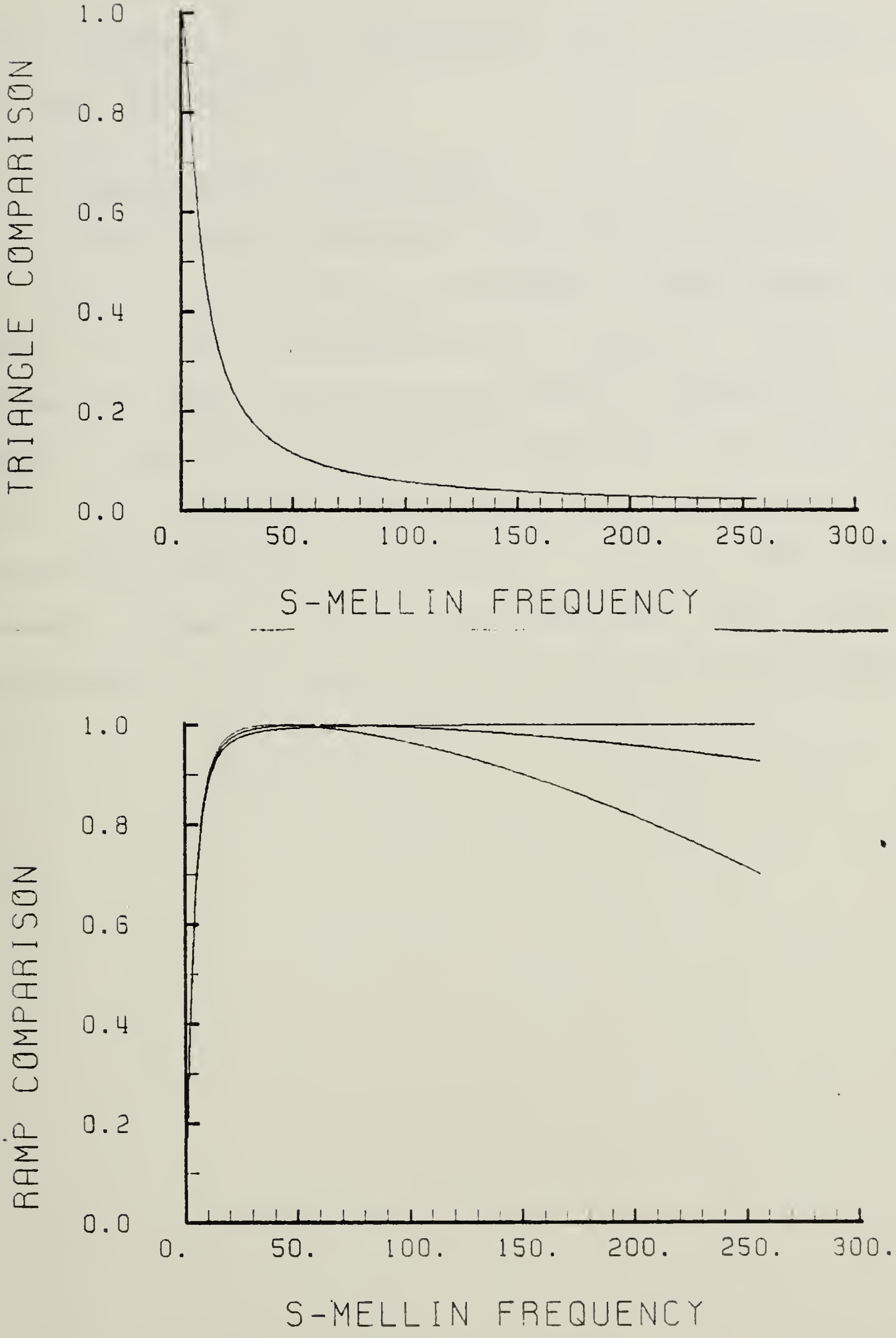
on second difference characteristics, but are modified by a 1/s term which has a stabilizing effect. This is equivalent to a division by x and integration in the x-domain. The order of the approximation has been increased by the modifier. Results using equation (60) and (62) should be alike. Figure 15 is the results of using (62) compared to the closed form solution to the figure 13 test shape transforms. An improved performance over (60) is seen over some of the range, but a drop as frequency increases degrades the accuracy in figure 15b.

3. Higher Difference Approximations

Higher difference approximations can be developed. For instance, one algorithm depending upon the third difference is,

$$H_{a_3} \doteq \frac{1}{s^4} \sum_{n=1}^{N-2} \Delta^3 h_n \, n^{s+2} \tag{64}$$

The performance of higher order algorithms becomes increasingly suspect because of the extreme weighting they apply to different parts of the series. Different algorithms exist, but this weighting is always a factor. A smoother transform is achieved, but a large error is likely to develop due to the algorithm's dependence on higher order derivatives and the nature of sampled data. However,



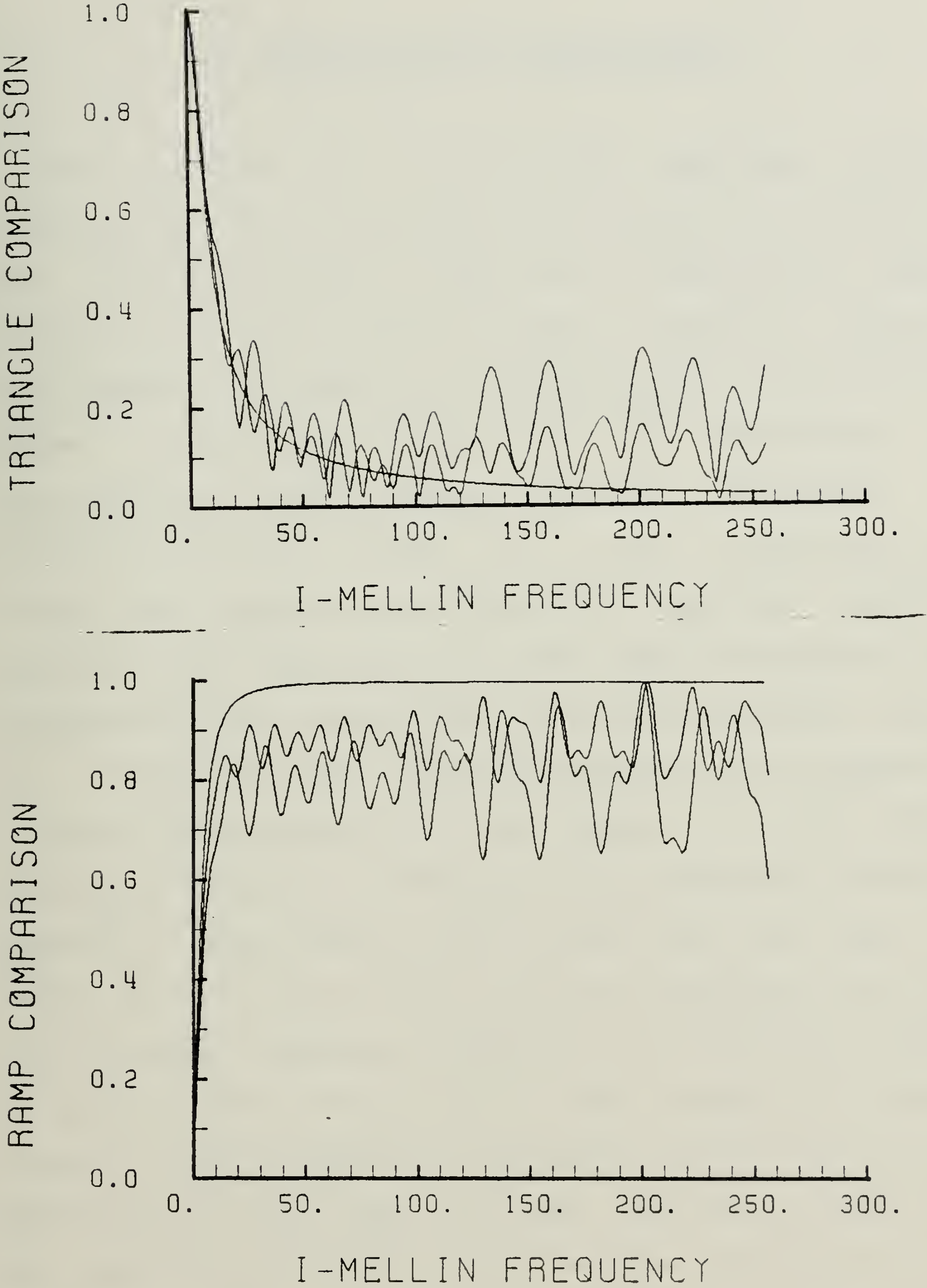
Second Difference Results

Figure 15

these comments are speculative since they were not confirmed experimentally.

4. Higher Order Integrations

Higher order integration rules should be able to be used with a corresponding improvement in performance. One using the first difference with Simpson´s rule was implemented with dissappointing results. Figure 16 is the result of such an implementation. The droop for the ramp input is apparent even though not present in the trapezoidal rule used in subsection 1 above. The higher frequency error is also more prevalent than before. A program error is of course suspected, but was never found.



Simpson's Rule Results

Figure 16

IV. CLASSIFICATION PREPROCESSING

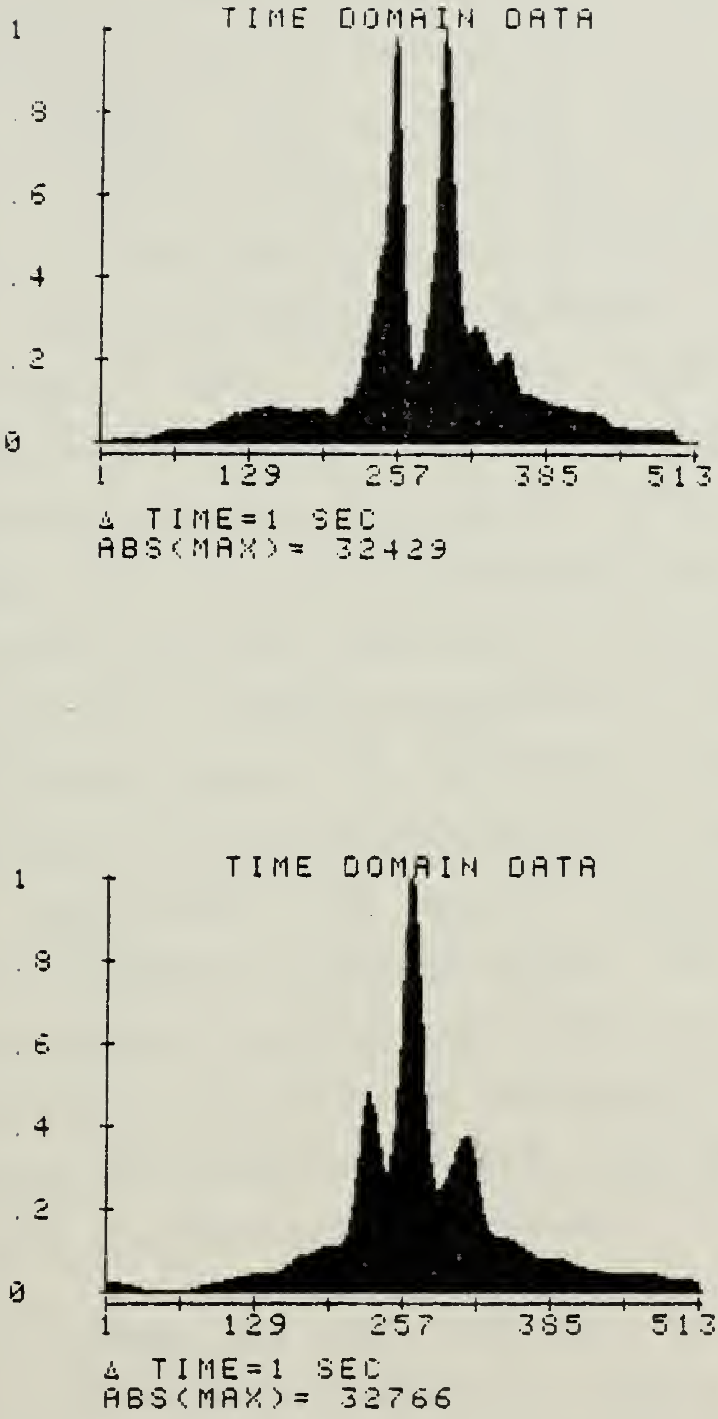
The problem of determining important signal characterisics can be approached in at least two different ways. First, by trying to learn what is important in human recognition, and then trying to adapt a machine to emulate that behavior. Or second, by using successive transforms to remove information known to be superfluous to classification, while keeping enough information to reliably assign an object to a class. Addressing the former, even though it is difficult to determine specific details, some key aspects of human visual recognition are discernable. Chief among these is that the intensity level of a scene, or object, does not appear to be as important as the relative position of the edges, i.e. the shape separating different intensities and frequencies [25-28]. Examples in scene analysis show clearly that the edges or shapes are far more critical to human recognition than the relative power differences themselves. The invariant shapes or angles in scene analysis find their analog in ratios between similiar points in differently scaled time series. Examination of many range only radar video ship signatures has provided a basis for noting that the relative position of time domain features remain constant over changes in aspect angle, while the relative intensity of the features

vary greatly as shown in figure 17. As the aspect changes, if this set of ratios remains constant within a ship class, then the set is identified as information worth preserving for the classifier. Conversely, since relative intensitiy is not a stable measure over aspect angle, or from among different ships of one class, that information should be intentionally removed to provide tighter natural clustering, with the minimum number of features.

A. INFORMATION REQUIRED TO CLASSIFY

There are two preconditions that must both exist for a set of possible input signatures to separate into distinct classes. First, the features of a particular class must have some common characteristic about them, and second, this characteristic must in some way be unique with respect to other classes. The assumption in existing radar signature classification projects is that there is enough information in the signatures to permit this classification. Short of actually trying to classify with a set of realistic signatures, the analytical determination that sufficient information is present in a set of all possible ship signatures is difficult to approach.

To establish how well the autonomous classifier is performing some measure of the classifiability of the set



Ship Signatures 10 Degrees Apart

Figure 17

received signals is desired. Failing this, some discussion of the information capacity of the preprocessor should at least be considered. The preprocessor produces the FFT magnitude of a sampled signal as the output of the first stage. Most of the unique positional relationships of the signal upon which human recognition apparently depend has been destroyed. Next, the Mellin transfrom stage effectively distorts the signal and uses the magnitude of a second Fourier transform as the output features. Signals reconstructed on the basis of FFT phase information alone usually provide sufficient similarity to be associated with the original signal, whereas reconstruction on the basis of magnitude does not retain any significant detail except when the signal is symmetric [29]. The data is also masked. For most applications, the data is frequency shifted, filtered and sampled as a baseband signal. This windowing masks the magnitude characteristic and is specifically designed into the processing. The resulting FM features are insensitive to positional and scaling relationships that are necessary in human visual recognition.

Analytic support is also available to quantify the importance of relative position of events [29]. By considering rms error (due to spectral phase and amplitude quantization for random signals) it has been concluded that approximately two more bits are required for the

quantization of phase information than amplitude for the same rms error. A separate analysis applied distortion rate theory to real-part, imaginary- part, and magnitude-phase encoding of the DFT of random sequences. The result was that phase required 1.4 bits more storage than magnitude for a similar error [30]. A third approach concluded that the Fourier phase includes 1.8 bits more information than the magnitude [31]. This was based on analysis of image reconstruction from kinoforms (phase-only holograms). The fact that phase-only reconstruction preserves much of the correlation between signals would suggest that the location of events tends to be preserved. Further, it seems that this information is lost by taking only the magnitude of the Fourier transform. Another interesting, albeit informal, view is apparent as one considers the phase-only signal as a spectral whitening process.

$$\mathcal{F}[f(t)] = F(f) = |F(f)|\, e^{-j\theta_f} \tag{65}$$

For reconstruction by phase alone, where the magnitude is set to one;

$$\mathcal{F}[f_p(t)] = \frac{1}{|F(f)|}\,\mathcal{F}[f(t)] \tag{66}$$

Since the received radar signature will have an abundance of low frequency spectral lines and smaller high frequency components, the low frequency information is not weighted

as heavily as the higher frequency information in the phase reconstructed signal. This seems like it would accentuate sharp changes in the reconstructed signature without removing relative location information. The result is the summation of the different frequency components, all with zero phase (i.e., no positional or amplitude distribution information can remain).

Considering the Mellin transform next, in a continuous case, the exponential warp does not lose any information. To zero the first data sample as required by the Chapter III modifications, will surely destroy information, but this may be confined to the DC term alone. The information lost during the final transform and magnitude is difficult to assess. Increased masking occurs due to the spectral truncation, so actual information loss may not be as great, but masking distortion may be greater than before. So approximately two bits of information are lost. Only a quarter of what was, remains. Information is also lost when the transforms are normalized in the processing so that any power calculation is also meaningless. Some interesting questions arise. After removing positional relationships, scaling, and power, what signal qualities remain and are they useful in classification? Although it is true that this insensitivity may add a certain robustness to the

system, the arbitrary loss of valid classification information should be minimized.

By examining the quality that must be ignored, and by comparing its removal to what is actually removed by the processing, an interesting result will develop. The effect of a time domain shift on the frequency domain is an additive phase term, linearly related to the frequncy of the coefficient, as in equation (2). Most of the structure of the signal is held in the phase relationships with respect to the fundamental and higher frequency terms. So shift can be defined as the phase of the fundamental complex coefficient. By setting the phase to zero, and adjusting the other coefficients according to their component frequency, the structure of the signal is not lost but reconstructed about the fundamental as before. If there are N/2 spectral phase angles, only the fundamental needs to be zeroed to remove the shift. If the information is contained uniformly in the spectral phase, then only 2/N of this structural information needs to be removed. When the magnitude is taken to produce shift invarient features, all of the phase relationships are destroyed. (N-2)/N of the information once held in the phase was removed needlessly. The amount of information lost removing the shift can be made arbitrarily small. As N grows unbounded, the amount of information that needs to be removed tends to

zero. This result seems to be supported by experience. With enough samples of a shape, its position with respect to the observation field is immaterial. In theory the principle is sound, but some practical limitations may degrade predicted performance. Recalling that an exponential warp translates scaling to shifting generalizes the result a bit further.

$$h(t/k) = h(e^{x-\ln k}) \tag{67}$$

The same priciple that permits simple shift removal is also valid for the removal of scaling dependence as well. Using the Mellin transform, scaling dependence may be removed by zeroing the fundamental and adjusting all the other coefficients as described above. For the FM preprocessor, the information lost removing the scaling and shifting dependence may be made arbitrarily small by increasing the number of spectral samples used. Since the number of spectral samples can be increased by filling zeros onto the finite signal in the original domain, this process does not effect the data sample rate. This approach was not verified experimentally, but represents a potentially powerful tool to analyse and improve the extracted feature space.

B. RANGE ONLY RADAR

This section addresses ship classification on the basis of information gathered from a range only radar video ship signature. An example of such a signature has already been considered as figure 16. Classification by range only radar signatures is subject to the same distortions discussed above. Typically, the radar return is detected and isolated in a range gate that is sampled and digitized. The rectangular sampling window can be considered the range gate itself. The range gate is designed to ensure that the included range is greater than the maximum ship length so that the time/range windowing has no effect on the frequency spectrum of the signature, other than increased spectral resolution. The placement of the ship signature in the window is not set, neither from encounter to encounter, nor from pulse to pulse (jitter). The total effect joins together to produce the framing distortions. Signature scaling results from viewing the ship from different aspect angles. The sampling rate must be done at more that twice the inverse of the resolution of the receiver. Quantization levels are chosen in a manner to reduce that predictable random noise to an acceptable level. The pulse to pulse jitter is an ever present characteristic of the radar problem, but at a normal resolution (greater than 25 feet) integrating the return in the time domain removes the

jitter effect, reduces scintillation, and improves the resolution of the signature. Although predetection (coherent) integration is more efficient, post detection (noncoherent) integration is more common because of the convenience of not having to preserve the radar frequency (RF) phase. For post detection integration of n pulses, the signal to noise ratio would be something less than n times the signal to noise ratio for one pulse [32]. More important to the recognition problem itself, for a stable system by the law of large numbers [33], fluctuation of the average value of the return will be overcome. That is, with the integration of n pulses the resolution (R) will become finer as

$$R = \sqrt{S^2/Dn} \qquad (68)$$

with a probability of 1-D, where S squared is the variance of the signal from pulse to pulse. For very high resolution, the cost of making the signature stable with respect to the integrator becomes prohibitive. It has become convenient to integrate the spectrum of the signature because the jitter effects can be completely removed. Another limiting factor in integrating over a period of time is that the position and aspect of the target are dynamic. They change with time. A conceptually attractive solution is a recursive filter which weights the

integrated pulses such that the older they become, the less weight is accorded to them.

If the course and speed of the target ship are known from measurement of the target track, it is possible to infer the aspect of the ship. The range profile can give some estimation of size. Unfortunately, the three dimensional change in aspect angle, commonly suffered by a ship presents more than just a video signature scaling change. The radar cross section of even individual structural components of the radar target changes with respect to the aspect angle. The composite effect is that ship signatures vary greatly with aspect angle. The radar is an electromagnetic sensor, reacting to energy reflected from the target. These reflections are a result of scatterers that are related in dimension to the wavelength of the illuminating energy. Because of the great difference in wavelength between light and microwaves, what can be "seen" by radar may be quite different than that seen by an eye. Also, when measuring size or any distances with a radar of high resolution (less than 50 feet), an error can be incurred since the extremities of the target are not always good scatterers. Echoes from the forward or stern portions of the target might be observed in the noise, especially for a relatively low power radar [32]. After reviewing hundreds of signatures, it appears that major

features, such as overall ship length and dominant mast structure are frequently discernable, but vary in relative amplitude. Resonance, shadowing (one reflector hiding another), multipath returns, the mapping of three dimensional aspect changes onto a one dimensional time series, and the amplitude and phase of component returns summing constructively or destructively to cause a scintillation of the composite target. Some of the variations caused by these conditions can be lessened by integration but major effects remain causing the signature to vary in shape and content with the aspect angle. For this reason, a class feature volume cannot be reduced to a single point, but will remain a hypervolume in the feature space even in the ideal case. Any selection of features should try to minimize this volume. Features should be selected that are relatively insensitive to known superfluous effects.

C. CLASS DISCRIMINATION

In the last chapter, the major concern was removing two sources of variance with no classifying value. Algorithms were developed and canonic shapes generated to verify the algorithms and demonstrate the invariance to scaling. In the same manner, this chapter has reviewed information loss and process masking. The question of whether sufficient

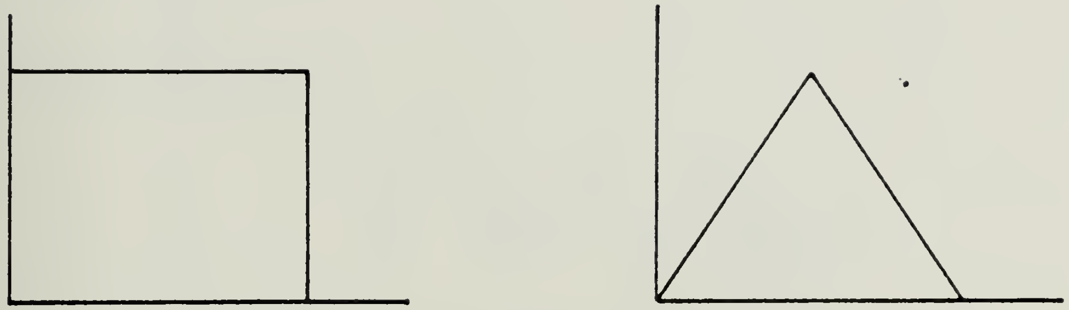
information is present to classify was raised. To answer this question some simple classes of canonic figures are defined, and put through the entire FM preprocessor documented in Appendix B. The DMT algorithm found to offer the best scale factor rejection in Chapter III was used to generate the final features. The algorithm chosen is based on a second difference modification and is defined in equation (62),

$$H_{a2}(s) \doteq \frac{1}{s+1} \sum_{n=1}^{N-1} \Delta^2 h_n \, n^{s+1} \tag{62}$$

After canonic tests were made, preprocessor performance on several ship signatures was recorded. Although this was premature in the logical test sequence, the results are of some interest.

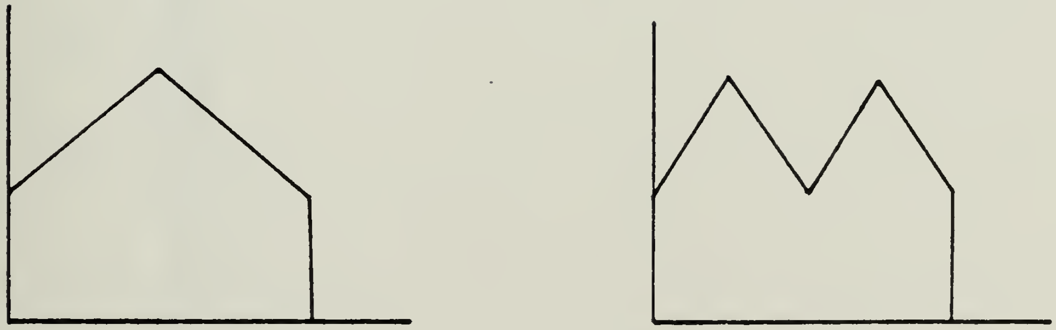
1. Test Shapes and Results

Four test shapes were used. Figure 18 shows the test shapes. All were scaled and shifted originally to test for algorithm verification and demonstrate scale invariance. In this series of tests they were left fixed and used in different combinations to try to detect shape presence in the FM feature space. The object is to differentiate between different canonic classes. Figure 19 shows a comparison of the shapes, rectangle and triangle. A test combining the rectangle and triangle was the subject of



a. RECT

b. TRI

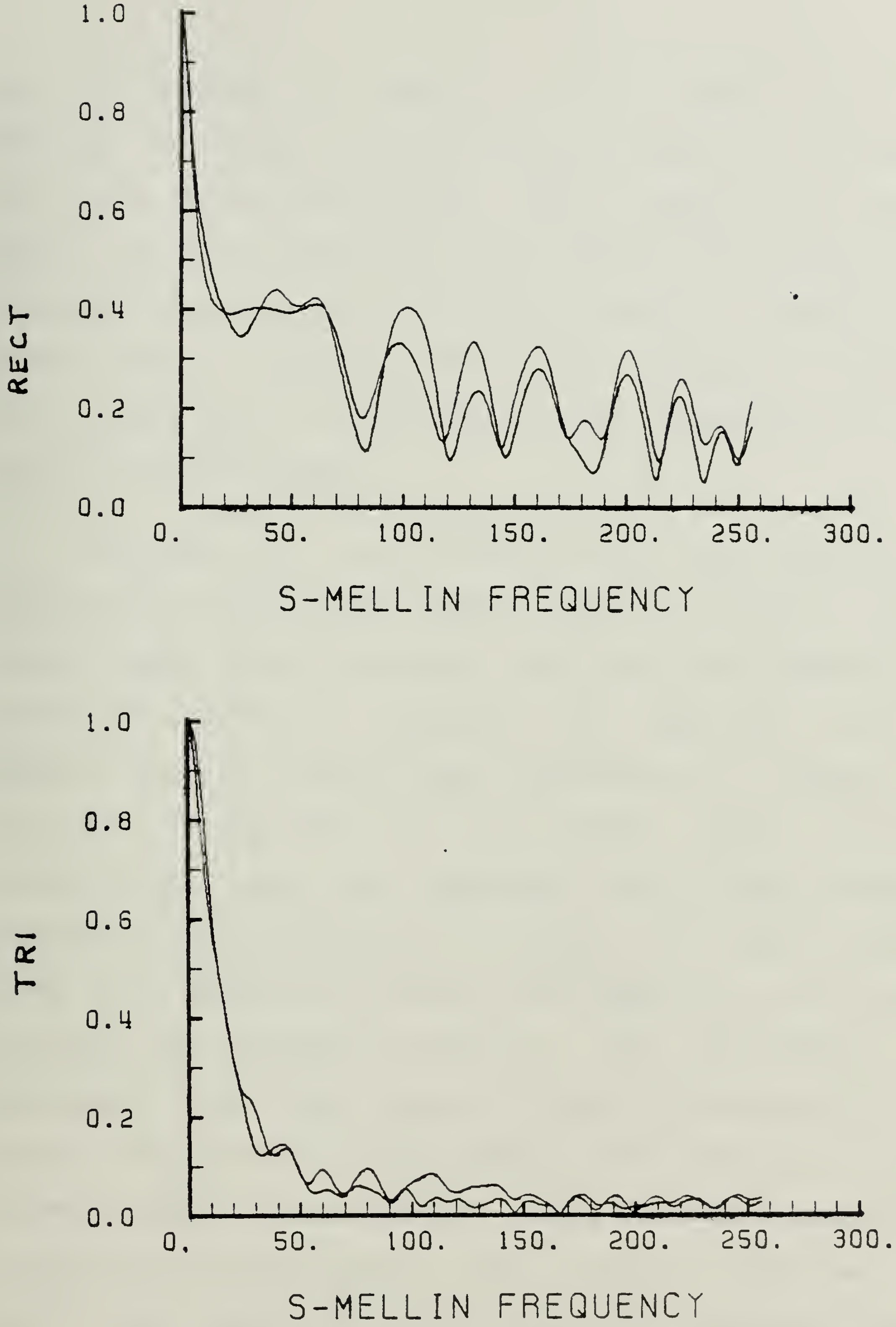


c. SHAPE 1

e. SHAPE 2

Test Shapes

Figure 18

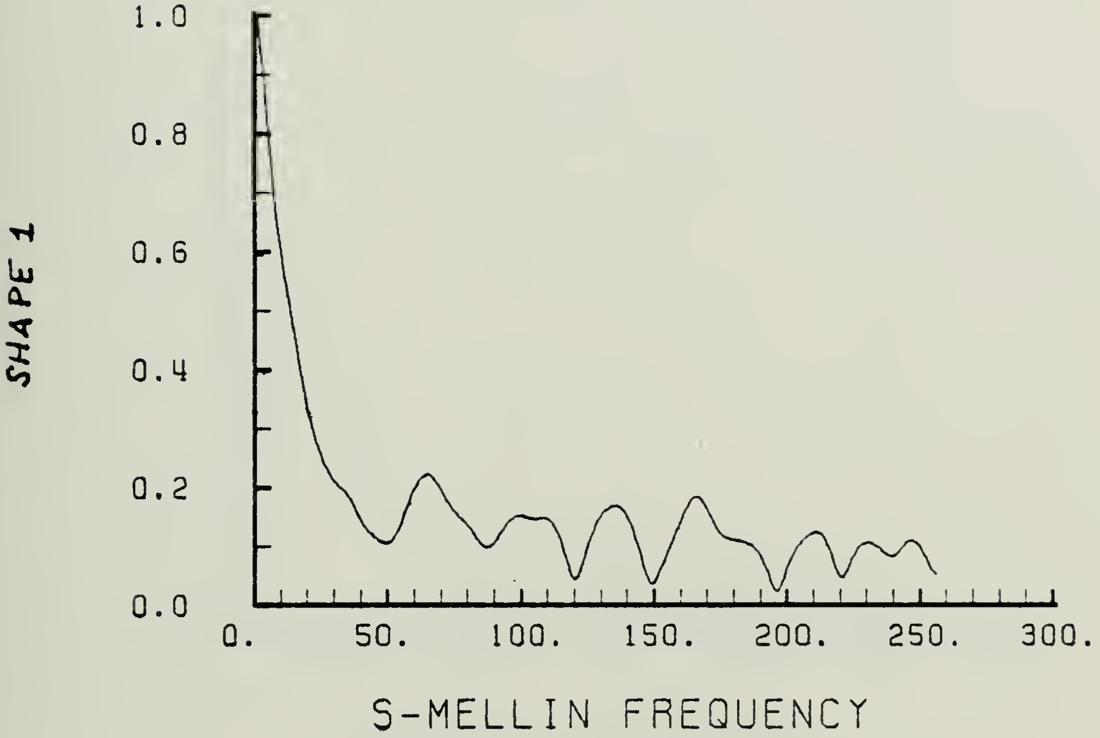


RECT and TRI FM Features

Figure 19

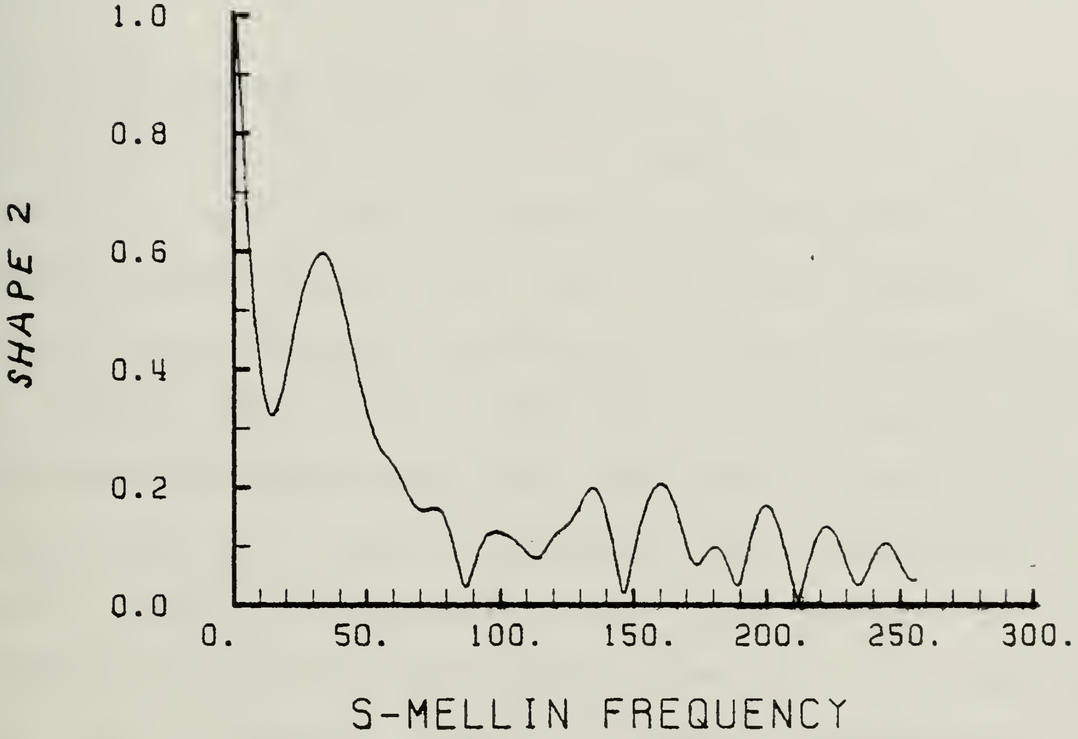
figure 20. Finally in figure 21 a rectangle with two triangles is shown in the FM feature space. It is clear from the plots that most of the scale variance has been removed. Just as important, some quality does remain that differentiates between the canonic shapes. A square in general can be differentiated from a triangle. A "ship" with a single mass of superstructure can be separated from one with two such masses.

Although it's clear from the plots that there is a unique quality left in the feature space to allow the time domain shapes to be classified, some quantified measure of system performance is required. The magnitude of the feeature vectors have all been normalized with respect to the first coefficient, so in that region little discrimination can be expected. For higher Mellin frequencies, noise dominates. A region of coefficients, 11-100 was chosen to classify the shapes by correlation. The results are included as Table 2. The improvement in performance over that shown by Table 1 in Chapter II is dramatic. The methods in that earlier test resulted in the observed length of an object being the distinguishing criterion for classification. Using methods supported in Chapter III, variations due to scaling and shifting of the original domain have been removed. The features now reflect the shape of the object in the time domain. An unusual



Shape 1 FM Features

Figure 20



Shape 2 FM Features

Figure 21

effect is noted with respect to the difference squared analysis in Table 2b. Although TRI2 is more closely correlated to TRI1 than RECT2 the squared error shows just the reverse. The results are encouraging. The preprocessor has greatly simplified the classification problem for the canonic shapes above.

2. Ship Signatures and Results

A single ship was used to make these preliminary tests for this thesis. Signatures were taken every ten degrees around a ship from zero to fifty degrees. The results are plotted and compared over twenty degree aspects in figures 22-23. The signatures are the result of very high resolution radar signature data that has been degraded and smoothed to a lower resolution with essentially no noise present. Recalling that the purpose of the preprocessor was to remove variance due to pure shifting and pure scaling, leaving enough information for classification, to the eye there seems to be little encouragement from these results. It is recalled that a goal of this preprocessor is to make the classifiers job easier by removing dependence on shifting and scaling of the original data. The information that remains depends on unspecified signal characteristics that here appear to useful in discriminating shape classes and possibly ship

classes. However no real conclusion can be drawn at this point because of the small data base and the absence of an automatic classifier to generate an optimal feature space.

Table 2

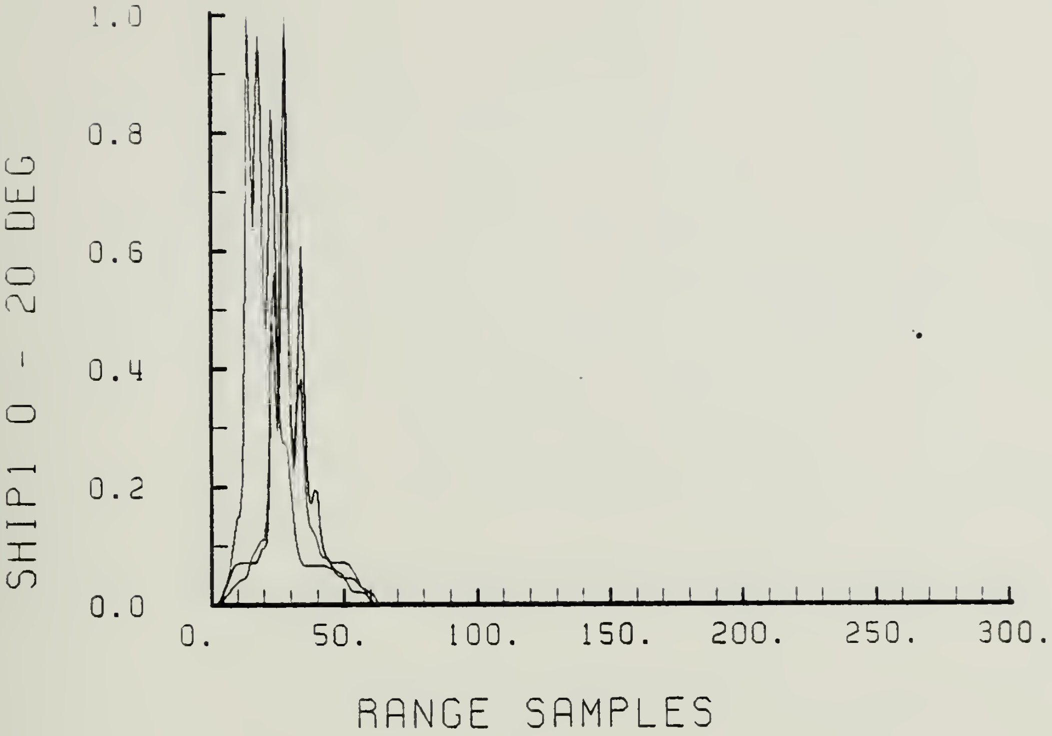
Canonic Shape Fourier - Mellin Feature Comparisons

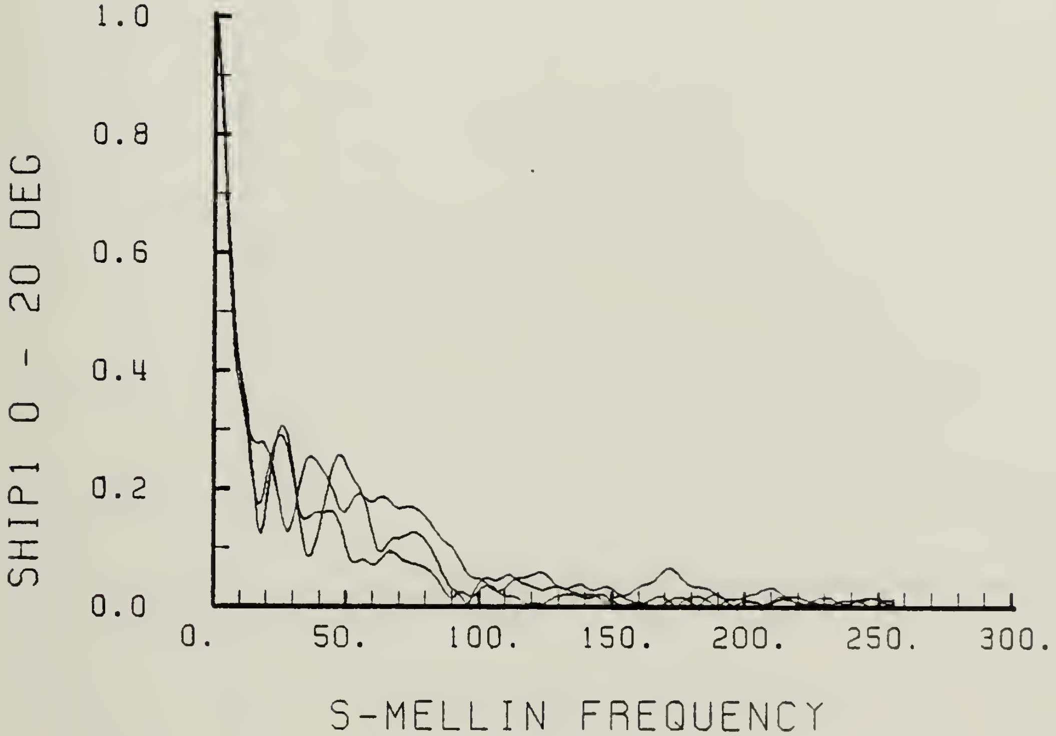
a. Peak Correlation Values

	RECT	RECT/2	TRI	TRI/2
RECT	1.00	0.95	0.50	0.42
RECT/2	-	1.00	0.52	0.41
TRI	-	-	1.00	0.98
TRI/2	-	-	-	1.00

b. Squared difference between features.

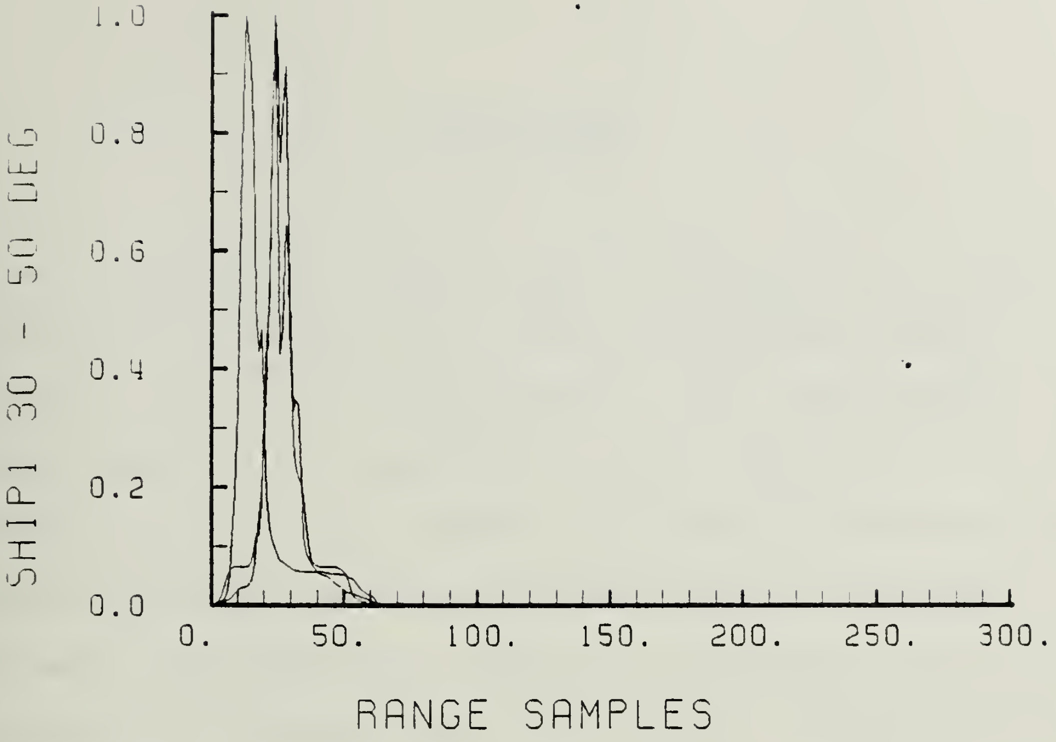
	RECT	RECT/2	TRI	TRI/2
RECT	.000	.011	.010	.011
RECT/2	-	.000	.015	.014
TRI	-	-	.000	.011
TRI/2	-	-	-	.000

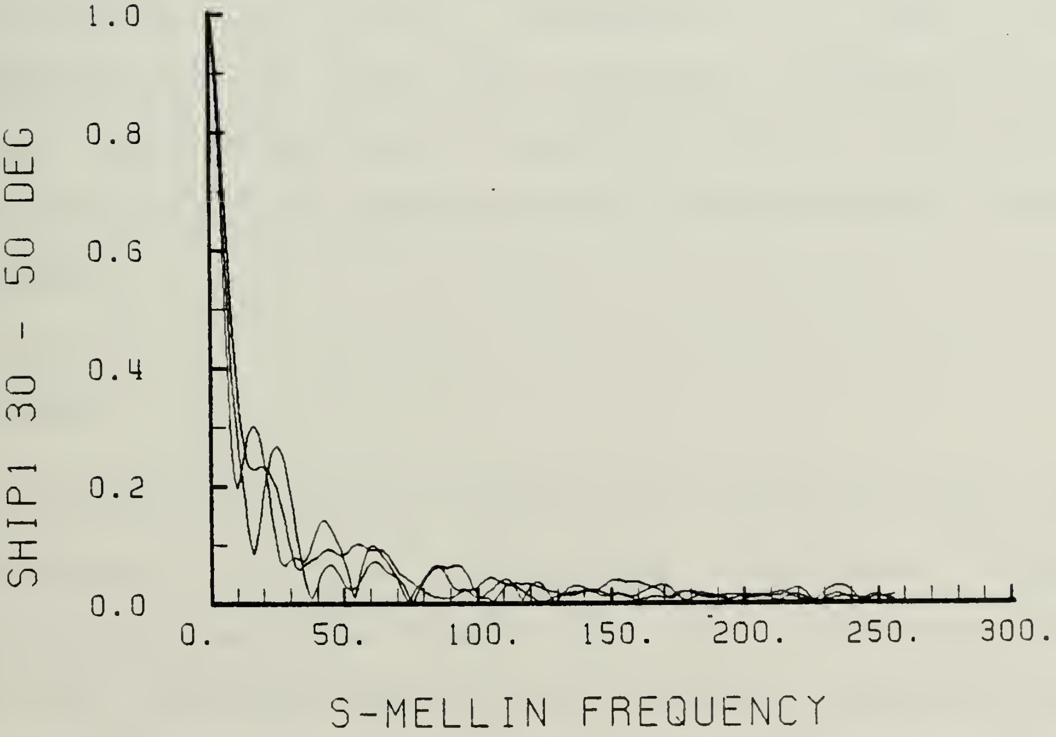




Ship 1 From 0 to 20 Degrees

Figure 22





Ship 1 From 30 to 50 Degrees

Figure 23

V. CONCLUSIONS

The preprocessor design began by considering a generic classification system. A distinction was drawn between the classifier and the preprocessor. The preprocessor is problem specific. It assists in the classification by extracting a set of features for the classifier. The extracted feature space has enhanced natural clustering on the basis of shape by removing information that was extraneous for classification. Two useless characteristics were identified as shifting and scaling. The preprocessor was designed to remove dependence on these two characteristics by using the invariant properties of a Fourier transform followed in series by a Mellin transform. The resulting set of coefficients are Fourier-Mellin (FM) features.

A. REVIEW

In Chapter II a Mellin transform was developed using the conventional digital processing approach which exponentially warps the domain and then transforms the spectrum by an FFT. This method is sometimes identified as a fast Mellin transform (FMT). The unbounded behavior of the exponentially warped frequency spectrum was shown to result in a function that could not be transformed. That

is, the warped function has no transform because the Mellin integral is indeterminate at the lower limit. Error correcting techniques cannot compensate for the effect because the error itself cannot be computed in general. The bound for the error was found and seen to be the envelope for existing error correction functions.

Chapter III developed some useful properties of the Mellin transform that were used to modify the signal so that the pitfalls isolated in Chapter II were avoided. This was done by modifying the input to the Mellin transform to always be transformable. To simplify the implementation and to control the effects of sampling, a direct Mellin transform was used for the development of the modifiers. Several suitable modifiers were determined and tested with differently scaled inputs for which the closed form solution was known. The direct Mellin algorithm that produced features closest to the closed form solution was chosen for use in the preprocessor. With the modifications in place, the preprocessor was tested and shown to produce features that were invariant to shifting and scaling. It was also shown that the features retained enough information to classify canonic shapes.

Chapter IV discussed what type of information is required for classification. Signal structure or shape was

identified as key information. A discussion of what information is required for classification and a means of keeping the signal structure intact throughout the preprocessor was advanced, but not empirically verified.

B. FUTURE WORK

The design FM preprocessor does produce a feature space with enhanced clustering, but problems remain to be resolved before the full potential of the system can be realized. There are three extant conditions that detract from the performance of the implemented preprocessor. First, the preprocessor is not computationally efficient. Second, most of the signal structure that should be vital to classification is obviously lost. And third, a complete verification of performance has not been conducted. The current preprocessor produces an enhanced feature base for a classifier, but attention to these main weak points will greatly improve the applied techniques.

1. Efficient Processing

A direct Mellin transform using N spectral samples to transform into M Mellin spectral coefficients requires M by N multiplications. If only a few coefficients are to be used then the number of multiplications may be small. Using the FMT requires about N(20+ln(N)) arithmetic operations

for the interpolation and the FFT. If twenty-five or more Mellin coefficients are required, the FMT is faster. The modifiers used to prepare the spectrum for the direct Mellin transform will also work for as FMT, but this should be demonstrated experimentally. Because the FMT directly weights the lower numbered samples it may produce more accurate results as well.

2. Conservation of Information

Chapter IV discussed the type of information required for visual pattern recognition in humans. The preservation of this information should be a specified design goal for the preprocessor. The preprocessors built for this thesis removed much of these vital signal properies. A means was introduced to limit or control the loss of structural detail by zeroing the fundamental phase and adjusting each of the remaining complex coefficients to reconstruct phase relationships. This may be done in the frequency domain as described, or by a similar operation in the time domain, shifting the centroid to zero. In either case, to transform more information about the signal will effectively increase the sensitivity of the features to characteristics in the time domain. This increase in sensitivity needs measured to confirm the approach. It is also possible that continued selective zeroing of

coefficients may offer improved performance or robustness to the system as a whole.

3. Verification

Although the improved performance was demonstrated with respect to ealier FM digital preprocessing, a direct improvement factor needs to be established. Time domain correlation should be used as a measure of original signal classifiability. This measure needs to be compared to the FM domain correlation recorded as Table 2 in Chapter IV. Next, the preprocessor or an improved version will have to be married to a classifier and realistic data used to evaluate its effect on the classification system. The preprocessor built was design to be used on-line. An on-line classifier needs to be built as well.

The systematic evaluation of ship profiles using FM features is still a requirement. For several ships, FM features must be singled out and plotted together as a function of aspect angle. The purposes are to establish a range of aspect over which classification may be possible, to evaluate changes of structural content as discussed in section two above, and to determine the beam signature as a classification "node". Although these purposes assist in the system design, the third is more of an operational necessity. All ship signatures will degrade to the beam

aspect "node" so that this hypervolume in the feature space is occupied in common. Therefore the beam condition must be detected and withheld from ever entering the classifier. Nor should beam signatures be used for classifier training. The beam "node" needs to be determined separate from the classifier. Initially this information can be plotted to examine feature behavior and confirm the methodology, automated methods will quickly follow. These analysis functions may never reside in the classification system itself, but must be a part of the tools used to develop a workable classification system.

APPENDIX A

```
C
C   ****************************************************
C   * THIS PROGRAM IS A DIGITAL IMPLEMENTATION OF      *
C   * A FAST FOURIER TRANSFORM FOLLOWED BY A MELLIN    *
C   * TRANSFORM.  THE DIGITAL IMPLEMENTATION OF THE    *
C   * MELLIN IS AS AN EXPONENTIALLY SAMPLED SPECTRUM   *
C   * THEN RUN THROUGH AN FFT AND CORRECTED FOR THE    *
C   * ERROR BY ANY ONE OF SEVERAL CORRECTION           *
C   * SUBROUTINES, CORCTX.                             *
C   ****************************************************
C
C
C        RECORDED PLOTS / PLOT NUMBER USING RECORD CALLS
C             -TIME FUNCTION (CONT) / 1
C             -SAMPLED TIME FUNCTION / 1
C             -FFT (CONT) / 2
C             -EXPONENTIALLY SAMPLED FFT / 2
C             -UNIFORMLY SAMPLED FFT / 3
C             -DISCRETE MELLIN FEATURES / 4
C
C
C     RECORDED PLOTS / PLOT NUMBER USING RECANG CALLS
C             -TIME FUNCTION  (CONT) / 1
C             -FFT UNIFORMLY SAMPLED / 2 (MAG & PHASE)
C             -MELLIN FEATURES / 3 (MAG & PHASE)
C
C
C
      DIMENSION XREAL(200),XIMAG(200),XINT(200),T(64),
     1PRT(200)
      ISCALE = 31
      MU=5
      M=2**MU
      NU=7
      N=2**NU
      CALL SAMP(XREAL,XIMAG,N)
      CALL RECORD(XREAL,XIMAG,N,ISCALE)
C
C     COMPUTE THE ACTUAL SAMPLE POINTS AS PROVIDED BY THE
C     SAMPLED VIDEO.
C
      NU=5
      N=2**NU
      CALL SAMP(XREAL,XIMAG,N)
      CALL RECORD(XREAL,XIMAG,N,ISCALE)

      ISCALE = 31
      LU=7
      L=2**LU
      DO 200 I=1,L
      IF(I.GT.N) GO TO 150
      XINT(I)=XREAL(I)
      PRT(I)=XIMAG(I)
      GO TO 200
150   XINT(I)=0.0
      PRT(I)=0.0
200   CONTINUE
```

```
      CALL FFT(XINT,PRT,L,LU)
      CALL FFT(XREAL,XIMAG,N,NU)
310   CALL RECORD(XREAL,XIMAG,N,ISCALE)
      CALL RECORD(XINT,PRT,L,ISCALE)
      ISCALE = 31
      DO 300 I=1,N
      XREAL(I)=SQRT(XREAL(I)**2+XIMAG(I)**2)
      XIMAG(I)=0.
300   CONTINUE
C
C
C     CALCULATE THE NEW SAMPLE POINTS AND INTERPOLATE
C     TC FIND THE EXPONENTIALLY SAMPLED VALUES.
C
      CALL NUPTS(XINT,M,N)
C
C     AFTER NEW INTERVALS CALCULATED AND STORED TEMPORARILY
C     IN VECTOR XINT, STORE FOR LATER PRINTING IN T VECTOR.
C
      DO 400 I=1,M
      PRT(I)=0.0
      T(I)=XINT(I)
400   CONTINUE
C
C     WITH THE NEW TIMES IN VECTOR XINT, AND THE CURRENT
C     SPECTRUM SAMPLES IN VECTCR XREAL, COMPUTE THE NEW
C     EXPONENTIAL SAMPLES AND ENTER THESE INTO VECTOR
C     XINT BY USING THE CHOSEN INTERPOLATION METHOD.
C     FINALLY, RECORD THE FIRST SAMPLE TO BE USED LATER
C     TO CORRECT THE MELLIN TRANSFORM FOR LOW FREQUENCY
C     LOST IN THIS EXPONENTIALLY SAMPLED TECHNIQUE.
C
      F0=XREAL(1)
399   CONTINUE
      CALL INTP2(XREAL,N,XINT,M)
      WRITE(2,401)M
401   FORMAT(I4)
      CALL SMAX(XINT,PRT,M,SCALE)
      DO 410 I=1,M
      XREAL(I)=XINT(I)
      XINT(I)=SCALE * XINT(I)
      XIMAG(I)=0
      WRITE(2,415)T(I),XINT(I)
415   FORMAT(F10.5,7X,F10.5)
410   CONTINUE
C
C     SUBMIT THESE EXPONENTIALLY SAMPLED VALUES TO THE
C     FINAL FFT BLOCK.
C
      CALL FFT(XREAL,XIMAG,M,MU)
C
C     APPLY THE CORRECTION TERM, FIND THE MAGNITUDE
C     OF THE NOW SCALE AND TIME INVARIENT FEATURES.
C
CCC   CALL CORCT1(XREAL,XIMAG,M,F0)
      CALL CORCT2(XREAL,XIMAG,M,F0)
      CALL RECORD(XREAL,XIMAG,M,ISCALE)
      STOP
      END
```

```
C     *******************************************
C     * SUBROUTINE FFT, GIVEN THE COMPLEX       *
C     * SAMPLES (2**NU=N), WILL RETURN  THE     *
C     * N COEFFIECIENTS USING A DFT.            *
C     *******************************************
C
      SUBROUTINE FFT(XREAL,XIMAG,N,NU)
      DIMENSION XREAL(N),XIMAG(N)
      N2=N/2
      NU1=NU-1
      K=0
      DO 100 L=1,NU
102   DO 101 I=1,N2
      P=IBITR(K/2**NU1,NU)
      ARG=6.283185*P/FLOAT(N)
      C=COS(ARG)
      S=SIN(ARG)
      K1=K+1
      K1N2=K1+N2
      TREAL=XREAL(K1N2)*C+XIMAG(K1N2)*S
      TIMAG=XIMAG(K1N2)*C-XREAL(K1N2)*S
      XREAL(K1N2)=XREAL(K1)-TREAL
      XIMAG(K1N2)=XIMAG(K1)-TIMAG
      XREAL(K1)=XREAL(K1)+TREAL
      XIMAG(K1)=XIMAG(K1)+TIMAG
101   K=K+1
      K=K+N2
      IF(K.LT.N) GO TO 102
      K=0
      NU1=NU1-1
100   N2=N2/2
      DO 103 K=1,N
      I=IBITR(K-1,NU)+1
      IF(I.LE.K)GO TO 103
      TREAL=XREAL(K)
      TIMAG=XIMAG(K)
      XREAL(K)=XREAL(I)
      XIMAG(K)=XIMAG(I)
      XREAL(I)=TREAL
      XIMAG(I)=TIMAG
103   CONTINUE
      RETURN
      END
C
C
C
C     FUNCTION IBITR
      FUNCTION IBITR(J,NU)
      J1=J
      IBITR=0
      DO 200 I=1,NU
      J2=J1/2
      IBITR=IBITR*2+(J1-2*J2)
200   J1=J2
      RETURN
      END
```

```
C  *********************************************************
C  * INTP2 IS A SECOND ORDER INTERPOLATION BASED ON A      *
C  * POLINOMIAL EQUATING TO FIRST AND SECOND DERIVATIVES   *
C  * APPROXIMATED BY CENTRAL DIFFERENCES. THE INPUT VECTOR *
C  * <X> CONTAINS UNIFORMLY SAMPLED DATA WITH UNITY        *
C  * BETWEEN CONSECUTIVE SAMPLES.  VECTOR <Y> IS INPUT     *
C  * WITH THE NEW SAMPLE TIMES AND OUTPUT WITH             *
C  * THE NEW SAMPLES. THERE ARE N UNIFORM SAMPLES IN <X>   *
C  * AND M WARPED SAMPLES IN <Y>.                          *
C  *********************************************************
C
C
      SUBROUTINE INTP2(X,N,Y,M)
      DIMENSION X3(3),X(N),Y(M)
C
C
C     CHOSE THE X SAMPLE TIME CLOSES TO THE WARPED
C     TIME HELD IN <Y>
C
      DO 60 IY=1,M
      DT = 100
      DO 10 IX=1,N
      IXTIME = IX - 1
      DTABS = ABS(DT)
      DIST = FLOAT(IXTIME) - Y(IY)
      DIST = ABS(DIST)
      IF (DTABS .LT. DIST) GO TO 25
      DT = Y(IY) - IXTIME
10    CONTINUE
25    IXTIME = IXTIME - 1
      J1 = -1
      DO 30 J=1,3
      J1 = J + IXTIME - 1
      X3(J) = X(J1)
30    CONTINUE
      TIME = Y(IY)
      Y(IY) = FCN(X3,TIME,IXTIME)
60    CONTINUE
      RETURN
      END
C
C     FCN IS THE INTERPOLATION RULE.
C
      FUNCTION FCN(X,T,I)
      DIMENSION X(3)
      TX = FLOAT(I)
C
C     COMPUTE THE COEFFICIENTS.
C
      A = (X(3) - 2.* X(2) + X(1)) / 2.
      B = (X(3) - X(1)) / 2.0 - 2.* A * TX
      C = X(2) - A * TX**2. - B * TX
C
C     COMPUTE THE INTERPOLATED VALUE.
C
      FCN = A * (T**2) + B * T + C
      RETURN
      END
```

```
C     ***************************************************
C     * SUBROUTINE INTERP USES A LAGRANGE THIRD         *
C     * ORDER METHOD OR A SECOND ORDER POLYNOMIAL       *
C     * TO COMPUTE THE INPUT SAMPLE WAVEFORM.           *
C     *                                                 *
C     * THE SUBROUTINE INPUTS ARE:                      *
C     *             X-SAMPLED WAVEFORM                  *
C     *             N-THE NUMBER OF X SAMPLES           *
C     *             Y-INPUT SAMPLE TIMES                *
C     *              -OUTPUT ITERPOLATED SAMPLES        *
C     *             M-THE NUMBER OF Y SAMPLES           *
C     ***************************************************
C
C
      SUBROUTINE INTERP(X,N,Y,M)
      DIMENSION X4(4),X(N),Y(M),SHFTX(156)
C
C     FIRST CHOOSE XTIME NEAREST EACH YTIME,
C     THEN COMPUTE THE INTERPOLATED VALUE AT THAT PT.
C
C
      DO 100 I=1,N
      I5=I+5
      IF(I.GE.(N-5)) GO TO 101
      SHFTX(I5)=X(I)
      GO TO 100
101   I5=I-(N-5)
      SHFTX(I5)=X(I)
100   CONTINUE
      DO 40 I=1,M
40    CONTINUE
      DO 60 IY=1,M
      YP=100
      DO 10 IX=1,N
      IXT=IX-1
      YPABS=ABS(YP)
      DIST=ABS(IXT-Y(IY))
      IF(YPABS.LT.DIST) GO TO 25
      YP=Y(IY)-IXT
      IX4=IX+5
10    CONTINUE
25    IX4=IX4-2
      DO 30 J=1,4
      X4(J)=SHFTX(IX4+J)
30    CONTINUE
      IM1=IX4+1
      I0=IX4+2
      IP1=IX4+3
      IP2=IX4+4
35    Y(IY)=YLAGR(X4,YP)
60    CONTINUE
      RETURN
      END
```

```
C
C     FUNCTION YLIN MAKES A LINEAR INTERPOLATION.
C
C
      FUNCTION YLIN(X4,YP)
      DIMENSION X4(4)
      IF(YP.LT.0.) GO TO 50
      YLIN=X4(2)+YP*(X4(3)-X4(2))
      RETURN
50    YLIN=X4(2)-YP*(X4(1)-X4(2))
      RETURN
      END
C
C
C
C     FUNCTION YLAGR COMPUTES THE LAGRANGE MULTIPLIERS
C     AND MAKES THE INTERPOLATION FOR A CHOSEN OFFSET
C     FROM THE CENTRAL SAMPLE X4(2)
C
C
      FUNCTION YLAGR(X4,YP)
      DIMENSION X4(4)
      CM1=-YP*(YP-1)*(YP-2)/6.0
      C0=(YP**2-1)*(YP-2)/2.0
      CP1=-YP*(YP+1)*(YP-2)/2.0
      CP2=YP*(YP**2-1)/6.0
      YLAGR=CM1*X4(1)+C0*X4(2)+CP1*X4(3)+CP2*X4(4)
      RETURN
      END
C
C   **********************************************************
C   * SUBROUTINE NUPTS CALCULATES THE M EXPONENTIAL SAMPLE  *
C   * POINTS FROM THE N UNIFORM SAMPLES OF THE EXISTING     *
C   * SPECTRUM IN PREPARATION FOR AN INTERPOLATION.  THIS   *
C   * EXPONENTIALLY SAMPLED SET OF POINTS ARE STORED IN     *
C   * THE INPUT VECTOR X.                                   *
C   **********************************************************
C
      SUBROUTINE NUPTS(X,M,N)
      DIMENSION X(156)
      UN=FLOAT(N)/2.0 + 1.0
      EM=FLOAT(M)-1.0
      DELZ=LOG(UN)/EM
      DO 100 I=1,M
      SI=FLOAT(I)-1.0
      VALUE = SI*DELZ
      X(I)=EXP(VALUE)
100   CONTINUE
      RETURN
      END
```

```
C     **********************************************************
C     * THIS IS A STUB PROVIDING A TEST SKIN RETURN            *
C     * TO SIMULATE A SKIN RETURN.  NORMALLY THIS SUBROUTINE *
C     * WILL ACTUALLY SAMPLE A SKIN RETURN.                    *
C     **********************************************************
C
        SUBROUTINE SAMP(XR,XI,N)
        DIMENSION XR(N),XI(N)
C
C       DESIGNATE SCALE FACTOR AND UNSCALED TIME SHIFT.
C
        SCALE = 16. / 32.
        SHIFT = 0.0 / 32.
        SCALE = 1.0/SCALE
        T0= .5 + SHIFT
        N1=N-1
        DO 100 I=1,N
        XR(I)=0.
        XI(I)=0.
C
C       CALCULATE THE TIME OF THE SAMPLE.
C
        SK= (FLOAT(I)-1.0)/FLOAT(N1)
C
C       BUILD THE TEST SKIN RETURN.
C       IN THIS (C 2-PER) CASE A DOUBLE PERIMID.
        TSCALE = (SK-T0)*SCALE
        IWFM = 1
        IF(IWFM .EQ. 1) GO TO 10
        IF(IWFM .EQ. 2) GO TO 50
10      W = 8. / 32.
        IF (TSCALE .LT. -W) GO TO 100
        IF (TSCALE .GT. +W) GO TO 100
        IF (TSCALE .LT. T0) GO TO 20
        XR(I) = (T0 - TSCALE) * 10.0
        GO TO 100
20      XR(I) = W * 10.0 - (TSCALE - T0) * 10.0
        GO TO 100
50      CONTINUE
C       THE FOLLOWING WAVE FORM IS A SQUARE WAVE 16 SAMPLES
C       WIDE CENTERED AROUND THE SIXTEENTH SAMPLE.
C       SCALING AND SHIFTING DONE ABOVE WILL EFFECT THE
C       WAVEFORM ACCORDINGLY.


        TSCALE = (SK-T0)*SCALE
        EDGE = 1. / 4.
        IF (TSCALE .LT. -EDGE) GO TO 100
        IF (TSCALE .GT. +EDGE) GO TO 100
        XR(I) = 1.0
100     CONTINUE
        RETURN
        END
```

```
C   *********************************************************
C   * THIS SUBROUTINE RECORDS A SELECTED DATA SET,         *
C   * AND SCALES THE INDEX TO 32 SAMPLES (0-31).           *
C   * THE INPUTS ARE XREAL, XIMAG, AND N (THE              *
C   * NUMBER OF SAMPLES). ISCALE IS A CONTROL              *
C   * VARIABLE.                                            *
C   *      IF ISCALE = 0 NO SCALING IS PERFORMED           *
C   *              SCALE = 1, AXIS = 31                    *
C   *              SHIFT = 1 (STARTS AT T = 0 )            *
C   *         ISCALE < 0 SHIFTING AND SCALING OCCURS       *
C   *              |ISCALE| = AXIS LEN. A SHIFT IS INCORP- *
C   *              ERATED SO THAT AXIS STARTS AT 1.        *
C   *         ISCALE > 0 AXIS SCALING OCCURS BUT THERE IS  *
C   *              NO SHIFT (IE., THE AXIS STARTS AT 0.)   *
C   *                                                      *
C   *********************************************************
C
      SUBROUTINE RECORD(XR,XI,N,ISCALE)
      DIMENSION XR(N),XI(N),REC(200)
      SCALE=1.0
      AXIS = 31
      SHIFT = 1
      WRITE(2,50)N
50    FORMAT(I4)
      IF(ISCALE .EQ. 0) GO TO 40
      AXIS = FLOAT(ISCALE)
      AXIS = ABS(AXIS)
      IF( ISCALE .LT. 0) SHIFT = 0
      CALL SMAX(XR,XI,N,SCALE)
40    DO 100 I=1,N
      SI=(I-SHIFT)*AXIS/(FLOAT(N)-1.0)
      REC(I)=SQRT(XI(I)**2+XR(I)**2)
60    CONTINUE
      REC(I)=SCALE*REC(I)
      WRITE(2,75)SI,REC(I)
75    FORMAT(F10.5,7X,F10.5)
100   CONTINUE
      RETURN
      END
C
C
      SUBROUTINE SMAX(XR,XI,N,SCALE)
      DIMENSION XR(N),XI(N),T(200)
      XMAX=1.0
      DO 100 I=1,N
      T(I)=SQRT(XR(I)**2+XI(I)**2)
      IF(T(I).LT.XMAX) GO TO 100
      XMAX = T(I)
      SCALE=1.0/XMAX
100   CONTINUE
      RETURN
      END
```

```
C     *****************************************************
C     * THIS CORRECTION SUBROUTINE USES ONE OF            *
C     * THE SIMPLER CORRECTIONS FOR THE MELLIN            *
C     * TRANSFORM. THE CORRECTION IS A PURE IMAGINARY.    *
C     *                                                   *
C     *    CORRECTION = -F0/OMEGA                         *
C     *                                                   *
C     * THEN A MODIFICATION IS MADE TO THE ENTIRE         *
C     * TRANSFORM BY A MULTIPLICATION BY OMEGA.           *
C     *****************************************************

      SUBROUTINE CORCT1(XR,XI,N,F0)
      DIMENSION XR(N),XI(N)

      DO 100 I=1,N
      OMEGA = FLOAT(I) - 1.

      XR(I) = XR(I) * OMEGA
      IF(I .EQ. 1)GO TO 90
      XI(I) = (XI(I) - F0/OMEGA) * OMEGA
      GO TO 100
90    XI(I) = F0

100   CONTINUE

      RETURN
      END


C     ********************************************************
C     * THIS CORRECTION APPLIES THE MORE COMPLECATED         *
C     * EXPRESSION,                                          *
C     *    CORRECTION = F0/2 + JCOT(F0/OMEGA)                *
C     *                                                      *
C     * AND THEN MODIFIES THE ENTIRE TRANSFORM BY 1/OMEGA.   *
C     ********************************************************
C

      SUBROUTINE CORCT2(XR,XI,N,F0)
      DIMENSION XR(N),XI(N)

      DO 100 I=1,N
      OMEGA = FLOAT(I) - 1.

      XR(I) = (XR(I) + F0/2.) * OMEGA

      IF(I .EQ. 1)GO TO 90
      XI(I) = (XI(I) - (F0/2.)*COTAN(OMEGA))*OMEGA
      GO TO 100

90    XI(I) = F0/2.

100   CONTINUE

      RETURN
      END
```

APPENDIX B

```
C
C   *********************************************************
C   * THIS PROGRAM, FOURIER DIRECT MELLIN, TAKES AN INPUT    *
C   * WAVEFORM FROM LOGICAL DEVICE 2, PERFORMS AN FFT        *
C   * FOLLOWED BY A MDMT OUTPUTING THE FEATURES TO LOGICAL   *
C   * DEVICE 3 FOR LATER PLOTTING BY MELPLT.  THE MAJOR      *
C   * DATA STRUCTURES AND SUBROUTINES ARE LISTED BELOW       *
C   * IN ORDER OF THEIR APPEARANCE.  THE SUBROUTINES ARE     *
C   * DESCRIBED IN MORE DETAIL WHERE THEY ARE ACTUALLY       *
C   * LISTED IN THE PROGRAM.                                 *
C   *********************************************************
C
C
C
C
C        MAJOR DATA STRUCTURES:
C             <WFM> - THE INPUT WAVEFORM (REAL)
C             <XFM> - THE MELLIN TRANSFORM (COMPLEX)
C             <CPHI> - REAL MELLIN COEFFICIENTS
C             <SPHI> - IMAGINARY MELLIN COEFFICIENTS
C             <STAND> - AN ARRAY HELD FOR LATER COMPARISON
C                       OR OTHER USE.
C             <PRT> - AN ARRAY NORMALLY USED TO HOLD REAL
C                       DATA TEMPORILY. A WORK SPACE.
C             <IXT> - X AXIS TITLE FOR PLOTTING
C             <IYT> - Y AXIS TITLE FOR PLOTTING
C             <KEY> - NUMBER OF PLOT THIS GRAPH
C
C
C
C        SUBROUTINES:
C             WAVE - READS AN ARRAY FROM LOGICAL DEVICE 2,
C                  AND FILLS ZEROS TO MAKE A TOTAL OF 256
C                  SAMPLES. THE OUTPUT IS IN <WFM>.
C             FFT - AN FFT BLOCK
C             COEF - COMPUTES THE MELLIN TRANSFORM SAMPLE
C                       WEIGHTS. THESE ARE COMPLEX
C                       NUMBERS WHOSE REAL AND IMAGINARY
C                       PARTS ARE STORED IN <CPHI> AND
C                       <SPHI> RESPECTIVELY.
C             DMTM - APPLIES A MODIFIED DIRECT MELLIN
C                       TRANSFORM TO AN INPUT WAVEFORM
C                       PUTTING THE OUTPUT IN <XFM>.
C                       THE ALGORITHM IS BASED ON A FIRST
C                       BACKWARD DIFFERENCE.
C             SMT - APPLIES A MODIFIED MELLIN TRANSFORM
C                       BASED ON A SECOND DIFFERENCE.
C             SMT2 - APPLIES A MODIFIED MELLIN TRANSFORM,
C                       DIFFERENT THAN SMT, BUT ALSO BASED
C                       ON THE SECOND DIFFERENCE.
C             CDMT - APPLIES A MODIFIED MELLIN TRANSFORM
C                       JUST AS DMTM ABOVE, EXCEPT THAT THE
C                       CENTRAL DIFFERENCE IS USED.
```

```
C             SIMP - APPLIES A MODIFIED MELLIN TRANSFORM
C                       USING A BACKWARD DIFFERENCE AS IN
C                       DMTM, EXCEPT THAT THE INTEGRATION
C                       IS BY SIMPSON'S RULE INSTEAD OF THE
C                       TRAPEZOIDAL RULE.
C             XAB - TAKES THE MAGNITUDE OF THE COMPLEX
C                       TRANSFORM <XFM> AND PUTS THE
C                       MANITUDE IN A SPECIFIED VECTOR.
C             STOW - NORMALIZES A VECTOR BY ITS
C                       MAGNITUDE AND WRITES IT TO
C                       LOGICAL DEVICE 3 WITH A TITLE
C                       FROM LOGICAL DEVICE 4.
C             HOLD - LOADS ONE VECTOR INTO ANOTHER.
C             ALTER - CHANGES <WFM> BY SCALE &/OR SHIFT
C                       AND OUTPUTS TO A SPECIFIED ARRAY
C             INTP3 - A SECOND ORDER SPLINE INTERPOLATION.
C             CFORM - PROVIDES TWO CLOSED FORM SOLUTIONS
C                       FOR VERIFYING THE MELLIN ALGORITHMS.
C             TITLE - ENTITLES THE PLOTS ON THE BASIS OF
C                       THE CALLING PROGRAM.
C
C
```

```
C     SO THE MAIN PROGRAM STARTS!

      DIMENSION PRT(256),WFM(256),STAND(256),IF(5)
      COMMON XFM(256,2),CPHI(256,128),SPHI(256,128),PI,
     1IXT(10),IYT(10),KEY

      DATA IF/'  FR','EQUE','NCY ','    ','    '/

      PI = 3.141592654


C     HOW MANY WAVEFORMS ARE TO BE TRANSFORMED?
      READ(2,10) NUMWFM
 10   FORMAT(I4)


C     NTMS IS THE NUMBER OF TIME SAMPLES (INCLUDING
C     ANY ZERO FILLING).  IT IS A POWER OF TWO
C     FOR THE CONVENIENCE OF THE FFT.
C     MPTS IS THE NUMBER OF SAMPLES INPUT TO THE
C     MELLIN TRANSFORM BLOCK.  THE COEFFICIENTS
C     ARE COMPUTED NOW.
      NU = 8
      NTMS = 2**NU
      MPTS = NTMS/2
 50   FORMAT(I4)
      NCOEF = NTMS
      CALL COEF(NCOEF,MPTS)

C     SET UP THE LOOP FOR THE NUMBER OF WAVEFORMS
C     TO BE PROCESSED.
      DO 500 IWAVE=1,NUMWFM

C     GET THE NEXT INPUT WAVEFORM.
      CALL WAVE(WFM,NTMS)

C     ZERO THE <STAND> VECTOR TO BE USED AS THE
C     IMAGINARY PART OF THE NTMS TIME SAMPLES.
      DO 100 I=1,NTMS
      STAND(I) = 0.0
 100  CONTINUE

      CALL STOW(WFM,NTMS)
C     TAKE THE FFT.
      CALL TITLE(IF)
      CALL FFT(WFM,STAND,NTMS,NU)

C     TAKE THE MAGNITUDE AND PUT IT INTO THE COMMON
C     WAVEFORM <WFM>.
      DO 200 I=1,NTMS
      WFM(I) = WFM(I)**2 + STAND(I)**2
      WFM(I) = SQRT(WFM(I))
 200  CONTINUE
      CALL STOW(WFM,NTMS)

C     TAKE THE MELLIN TRANSFORM OF THIS SPECTRUM
C     USING THE FIRST HALF OF THE FFT SAMPLES,
C     OTHERWISE KNOWN AS MPTS=NTMS/2.  THESE ARE THE
C     ONLY UNIQUE VALUES.
      CALL DMTM(WFM,MPTS,NCOEF)
      CALL XAB(PRT,NCOEF)
      CALL STOW(PRT,NCOEF)
```

```
C
C
C      NEXT CALL THE SECOND ORDER RULE DMT
C      SUBROUTINES.  BOTH COMPUTE THE MELLIN
C      USING THE SECOND DIFFERENCE APPROXIMATION
C      INSTEAD OF THE FIRST DIFFERENCE APPROXIMATION
C      ABOVE.  OTHERWISE THE APPROACH IS THE SAME.
C
C


       CALL SMT(WFM,MPTS,NCOEF)
       CALL XAB(PRT,NCOEF)
       CALL STCW(PRT,NCOEF)

       CALL SMT2(WFM,MPTS,NCOEF)
       CALL XAB(PRT,NCOEF)
       CALL STCW(PRT,NCOEF)

C      CALL CDMT WHICH USES THE CENTRAL DIIFERENCE
C      RULE FOR APPROXIMATING THE TRANSFORM.

       CALL CDMT(WFM,MPTS,NCOEF)
       CALL XAB(PRT,NCOEF)
       CALL STCW(PRT,NCOEF)

C      CALL SIMP WHICH COMPUTES THE MELLIN TRANSFORM
C      USING THE FIRST DIFFERENCE ALGORITHM AND
C      SIMPSON'S RULE TO COMPUTE THE MODIFIED
C      MELLIN TRANSFORM.
C


       CALL SIMP(WFM,MPTS,NCOEF)
       CALL XAB(PRT,NCOEF)
       CALL STCW(PRT,NCOEF)

C      END THE TRANSFORM LOOP. THE TRANSFORM HAS BEEN
C      OUTPUT TO LOGICAL DEVICE 3 AND PREPARED WITH
C      TITLE INFORMATION PROVIDED BY LOGICAL DEVICE 2
C      FOR PLOTTING WITH PROGRAM MELPLT FORTRAN.
C      STAY IN THE LOOP IF MORE WAVEFORMS ARE AVAILABLE.
500    CONTINUE

       CALL CFORM(PRT,NCOEF)
       CALL CFORM(PRT,NCOEF)

       STOP
       END
```

```
C
C
C     ***************************************************
C     * THE WAVE SUBROUTINE PRODUCES A WAVEFORM, IN     *
C     * READ FROM THE LOGICAL UNIT 2, NORMALLY A SHIP   *
C     * DATA FILE.  THE N SAMPLES ARE READ AND THEN     *
C     * ZEROS ARE STUFFED TO FILL THE 256 SAMPLES.      *
C     * THE REQUESTED WAVEFORM IS OUTPUT IN THE         *
C     * COMMON ARRAY WFM(256).                          *
C     ***************************************************
C


      SUBROUTINE WAVE(WFM,NPTS)
      DIMENSION WFM(NPTS),ID(5)
      COMMON XFM(256,2),CPHI(256,128),SPHI(256,128),PI,
     1IXT(10),IYT(10),KEY
      DATA ID/' RAN','GE S','AMPL','ES  ','    '/

      READ(2,10)KEY
      READ(2,30) (IXT(I),I=1,10)
      READ(2,30) (IYT(I),I=1,10)
30    FORMAT(10A4)
50    READ(2,10)N
      READ(2,20) (WFM(I),I=1,N)
      CALL TITLE(ID)
10    FORMAT(I4)
20    FORMAT(F10.5)
      IF(N .EQ. NPTS) RETURN
      N = N + 1
      DO 100 I=N,NPTS
      WFM(I) = 0.0
100   CONTINUE
      RETURN
      END
```

```
C
C
C
C   *******************************************
C   * SUBROUTINE FFT, GIVEN THE COMPLEX       *
C   * SAMPLES (2**NU=N), WILL RETURN  THE     *
C   * N COEFFIECIENTS USING A FFT.            *
C   *******************************************
C
      SUBROUTINE FFT(XREAL,XIMAG,N,NU)
      DIMENSION XREAL(N),XIMAG(N)
      N2=N/2
      NU1=NU-1
      K=0
      DO 100 L=1,NU
102   DO 101 I=1,N2
      P=IBITR(K/2**NU1,NU)
      ARG=6.283185*P/FLOAT(N)
      C=COS(ARG)
      S=SIN(ARG)
      K1=K+1
      K1N2=K1+N2
      TREAL=XREAL(K1N2)*C+XIMAG(K1N2)*S
      TIMAG=XIMAG(K1N2)*C-XREAL(K1N2)*S
      XREAL(K1N2)=XREAL(K1)-TREAL
      XIMAG(K1N2)=XIMAG(K1)-TIMAG
      XREAL(K1)=XREAL(K1)+TREAL
      XIMAG(K1)=XIMAG(K1)+TIMAG
101   K=K+1
      K=K+N2
      IF(K.LT.N) GO TO 102
      K=0
      NU1=NU1-1
100   N2=N2/2
      DO 103 K=1,N
      I=IBITR(K-1,NU)+1
      IF(I.LE.K)GO TO 103
      TREAL=XREAL(K)
      TIMAG=XIMAG(K)
      XREAL(K)=XREAL(I)
      XIMAG(K)=XIMAG(I)
      XREAL(I)=TREAL
      XIMAG(I)=TIMAG
103   CONTINUE
      RETURN
      END
C
C
C
C
C
C     FUNCTION IBITR
      FUNCTION IBITR(J,NU)
      J1=J
      IBITR=0
      DO 200 I=1,NU
      J2=J1/2
      IBITR=IBITR*2+(J1-2*J2)
200   J1=J2
      RETURN
      END
```

```
C
C   **************************************************
C   * THE COEF SUBROUTINE COMPUTES THE MELLIN        *
C   * COEFFICIENTS IN TO COMMON ARRAYS CPHI AND      *
C   * SPHI THAT REPRESENT THE REAL AND IMAGINARY     *
C   * PARTS RESPECTIVELY. THE TERMS ARE COMPUTED     *
C   * BY THE FORMULA:                                *
C   *    PHI(I,J) = J**S  , WHERE S = A NORMALIZED   *
C   *                   DISCRETE RADIAN FREQUENCY.   *
C   **************************************************
C


      SUBROUTINE COEF(NCOEF,NPTS)
      COMMON XFM(256,2),CPHI(256,128),SPHI(256,128),PI,
     1IXT(10),IYT(10),KEY
      DO 100 I = 1,NCOEF
      RI = FLOAT(I)
      OMEGA = 2.0 * PI * RI / 36.
      DO 200 J = 1,NPTS
      RJ = FLOAT(J)
      CPHI(I,J) = COS(OMEGA * ALOG(RJ))
      SPHI(I,J) = SIN(OMEGA * ALOG(RJ))
200   CONTINUE
100   CONTINUE
      RETURN
      END
```

```
C
C
C  ********************************************************
C  *  THE DMTM SUBROUTINE PERFORMS A DESCRETE MELLIN      *
C  *  TRANSFORM ON THE ARRAY WFM.  THE FORMULA            *
C  *  FOR ONE MELLIN FREQUENCY VALUE IS:                  *
C  *      XFM(I)=SUM(K=1 TO NPTS) (WFM(I+1)-WFM(I))*K**S  *
C  *                                                      *
C  *  THE COMPLEX COMPONENTS CF K**S ARE COMPUTED PRIOR   *
C  *  TC CALLING DMTM AND STORED IN THE COMMON ARRAYS     *
C  *  CPHI AND SPHI AS REAL AND IMAGINARY PARTS           *
C  *  RESPECTIVELY. THE ALGORITHM IS BASED ON THE         *
C  *  FIRST DIFFERENCE. THE TRAPEZOIDAL RULE IS USED      *
C  *  FOR THE INTEGRATION.  THE COMPLEX OUTPUT FOR        *
C  *  THE TRANSFORM IN IN THE COMMON ARRAY <XFM>.         *
C  ********************************************************
C
C

      SUBROUTINE DMTM(SAMP,NPTS,NCOEF)
      DIMENSION SAMP(NPTS),ID(5)
      COMMON XFM(256,2),CPHI(256,128),SPHI(256,128),PI,
     1IXT(10),IYT(10),KEY
      DATA ID/'D-ME','LLIN',' FRE','QUEN','CY  '/

      CALL TITLE(ID)
      DC 100 I=1,NCOEF
      XFM(I,1)=0.0
      XFM(I,2)=0.0
      N1 = NPTS - 1
      DO 200 J=1,N1
      J1 = J + 1
      XFM(I,1) = CPHI(I,J) * (SAMP(J) - SAMP(J1)) + XFM(I,1)
      XFM(I,2) = SPHI(I,J) * (SAMP(J) - SAMP(J1)) + XFM(I,2)
200   CONTINUE
100   CCNTINUE
      RETURN
      END
```

```
C
C     *********************************************************
C     * SMT IS A SUBROUTINE THAT COMPUTES A NUMERICAL         *
C     * APPROXIMATICN TO THE MELLIN TRANSFORM AS DOES         *
C     * DMTM ABOVE.  SMT USES A TRAPAZOIDAL APPROXIMATION     *
C     * TC COMPUTE THE TRANSFORM, BUT USES THE SAME           *
C     * COEFFICIENT MATRIXES <CPHI> AND <SPHI> CONTAINED      *
C     * IN COMMON.                                            *
C     *********************************************************
C

      SUBROUTINE SMT(SAMP,NPTS,NCOEF)
      DIMENSICN SAMP(NPTS),ID(5)
      COMMON XFM(256,2),CPHI(256,128),SPHI(256,128),PI,
     1IXT(10),IYT(10),KEY
      DATA ID/'S-ME','LLIN',' FRE','QUEN','CY  '/
      CALL TITLE(ID)

C     INITIALIZE THE  INPUT ARRAY AND COMPUTE
C     THE LOOP CONSTANTS.
      N1 = NPTS - 1
      N2 = N1 - 1

C     SET UP THE TRANSFORM LOCP. THE OUTER LOOP
C     SETS UP THE COEFFIECIENTS WHILE THE INNER
C     LOOP COMPUTES THE SUM WHICH ARE THE
C     CCEFFICIENTS.

      DO 200 J = 1,NCOEF
      XFM(J,1) = 0.0
      XFM(J,2) = 0.0
      DC 100 I = 1,N2
      I0 = I
      I1 = I + 1
      I2 = I + 2
      DELTA = SAMP(I0) - 2.* SAMP(I1) + SAMP(I2)
      XFM(J,1) = XFM(J,1) + DELTA*CPHI(J,I)*I
      XFM(J,2) = XFM(J,2) + DELTA*SPHI(J,I)*I
100   CCNTINUE

C     XFM(J,1) = XFM(J,1) / (FLOAT(J)*PI/18.)
C     XFM(J,2) = XFM(J,2) / (FLOAT(J)*PI/18.)


      XFM(J,1) = XFM(J,1) / SQRT(1+(FLOAT(J)*PI/18.)**2)
      XFM(J,2) = XFM(J,2) / SQRT(1+(FLOAT(J)*PI/18.)**2)
200   CONTINUE
      RETURN
      END
```

```
C
C
C     ************************************************************
C     * SMT2 IS A SUBROUTINE THAT COMPUTES A NUMERICAL           *
C     * APPROXIMATION TO THE MELLIN TRANSFORM AS DOES            *
C     * DMTM ABOVE.  SMT USES A TRAPAZOIDAL APPROXIMATION        *
C     * TO COMPUTE THE TRANSFORM, BUT USES THE SAME              *
C     * COEFFICIENT MATRIXES <CPHI> AND <SPHI> CONTAINED         *
C     * IN COMMON.                                               *
C     ************************************************************
C


      SUBROUTINE SMT2(SAMP,NPTS,NCOEF)
      DIMENSION SAMP(NPTS),ID(5)
      COMMON XFM(256,2),CPHI(256,128),SPHI(256,128),PI,
     1IXT(10),IYT(10),KEY
      DATA ID/'S2-M','ELLI','N FR','EQUE','NCY '/
      CALL TITLE(ID)

C     INITIALIZE THE  INPUT ARRAY AND COMPUTE
C     THE LOOP CONSTANTS.
      N1 = NPTS - 1
      N2 = N1 - 1

C     SET UP THE TRANSFORM LOOP. THE OUTER LOOP
C     SETS UP THE COEFFIECIENTS WHILE THE INNER
C     LOOP COMPUTES THE SUM WHICH ARE THE
C     COEFFICIENTS.

      DO 200 J = 1,NCOEF
      XFM(J,1) = 0.0
      XFM(J,2) = 0.0
      DO 100 I = 1,N2
      I0 = I
      I1 = I + 1
      I2 = I + 2
      DELTA = I*(SAMP(I0) - 2.* SAMP(I1) + SAMP(I2))
      DELTA = DELTA + (SAMP(I2) - SAMP(I0))
      XFM(J,1) = XFM(J,1) + DELTA*CPHI(J,I)
      XFM(J,2) = XFM(J,2) + DELTA*SPHI(J,I)
100   CONTINUE

C     REMOVE THE COLOR
      XFM(J,1) = XFM(J,1) / SQRT(1+(FLOAT(J)*PI/18.)**2)
      XFM(J,2) = XFM(J,2) / SQRT(1+(FLOAT(J)*PI/18.)**2)
200   CONTINUE
      RETURN
      END
```

```
C
C
C  ********************************************************
C  *   THE CDMT SUBROUTINE PERFORMS A DESCRETE MELLIN     *
C  *   TRANSFORM ON THE COMMON ARRAY WFM.  THE FORMULA    *
C  *   FOR ONE MELLIN FREQUENCY VALUE IS:                 *
C  *     XFM(I)=SUM(K=1 TO NPTS)(WFM(I+1)-WFM(I-1))*K**S  *
C  *                                                      *
C  *   THE COMPLEX COMPONENTS OF K**S ARE COMPUTED PRIOR  *
C  *   TO CALLING CDMT AND STORED IN THE COMMON ARRAYS    *
C  *   CPHI AND SPHI AS REAL AND IMAGINARY PARTS          *
C  *   RESPECTIVELY. THE CENTRAL DIFFERENCE IS USED.      *
C  ********************************************************
C
C


      SUBROUTINE CDMT(H,NPTS,NCOEF)
      DIMENSION H(NPTS),ID(5)
      COMMON XFM(256,2),CPHI(256,128),SPHI(256,128),PI,
     1IXT(10),IYT(10),KEY
      DATA ID/'C-ME','LLIN',' FRE','QUEN','CY  '/
      CALL TITLE(ID)

C     SET UP DERIVATIVE OF INPUT VECTOR <H(I)>
C     IDENTIFIED HERE AS <G>.

      NG = NPTS - 1

      DO 200 J = 1 , NCOEF
      XFM(J,1) = 0.0
      XFM(J,2) = 0.0

      OMEGA = FLOAT(J) * PI / 18.

      DO 100 I = 1,NG
      IM1 = I
      IP1 = I + 2

      DH = (H(IP1) - H(IM1)) / 2.

C     COMPUTE THE J-TH COEFFICIENTS BY THE SUM.

      XFM(J,1) = XFM(J,1) + DH * CPHI(J,I)
      XFM(J,2) = XFM(J,2) + DH * SPHI(J,I)

100   CONTINUE
C
      COLOR = 1.
      XFM(J,1) = XFM(J,1) * COLOR
      XFM(J,2) = XFM(J,2) * COLOR

200   CONTINUE

      RETURN
      END
```

```
C     ********************************************************
C     * THIS SUBROUTINE USES SIMPSON'S RULE TO APPROXIMATE   *
C     * THE MODIFIED MELLIN TRANSFORM.  THE MODIFICATION     *
C     * IS FREQUENCY TIMES THE FREQUENCY DERIVATIVE OF       *
C     * THE FFT.                                             *
C     ********************************************************
C


      SUBROUTINE SIMP(H,NPTS,NCOEF)
      DIMENSION H(NPTS),ID(5)
      COMMON XFM(256,2),CPHI(256,128),SPHI(256,128),PI,
     1IXT(10),IYT(10),KEY
      DATA ID/'I-ME','LLIN',' FRE','QUEN','CY  '/
      CALL TITLE(ID)

      N1 = NPTS -1
      N2 = NPTS - 2

      DO 100 J=1,NCOEF
      FSIMP = 2.0
      OMEGA = PI * FLOAT(J) / 18.
      COLOR = 1.
      XFM(J,1) = 0.
      XFM(J,2) = 0.

      DO 200 I=1,N1
      IM1 = I
      I0 = 1 + I

      IF (FSIMP .GT. 3.) GO TO 67
      FSIMP = 4.
      GO TO 69
67    FSIMP = 2.
69    CONTINUE
      DELTA = H(I0) - H(IM1)
      XFM(J,1) = XFM(J,1) + FSIMP * DELTA * CPHI(J,IM1)
      XFM(J,2) = XFM(J,2) + FSIMP * DELTA * SPHI(J,IM1)

200   CONTINUE

100   CONTINUE
      RETURN
      END
```

```
C
C     **************************************************
C     * THE XAB SUBROUTINE TAKES THE MAGNITUDE OF THE     *
C     * COMMON COMPLEX ARRAY XFM(256,2) AND PLACES        *
C     * THESE VALUES IN OUTPUT VECTOR <XMAG> FOR LATER    *
C     * PRINTING.                                         *
C     **************************************************
C
C


      SUBROUTINE XAB(XMAG,NPTS)
      DIMENSION XMAG(NPTS)
      COMMON XFM(256,2),CPHI(256,128),SPHI(256,128),PI,
     1IXT(10),IYT(10),KEY
      DO 100 I=1,NPTS
      XMAG(I) = SQRT(XFM(I,1)**2+XFM(I,2)**2)
100   CONTINUE
      RETURN
      END
```

```
C
C     ************************************************************
C     * THE STOW SUBROUTINE STORES THE INPUT ARRAY PRT(NPTS)     *
C     * INTO THE LOGICAL DEVICE 2 FOR LATER USE IN PLOTTING.     *
C     * NPLOT  NUMBERS THE PLOTS FOR LATER IDENTIFICETION,       *
C     * AND A PLOT TITLE FOR THE MELLIN FREQUENCY IS ADDED       *
C     * FOR CONVENIENCE.                                         *
C     *       THE MODULUS OF THE                                 *
C     * TRANSFORM TAKEN, SCALED TO UNIT MAGNITUDE, AND           *
C     * OUTPUT TO LOGICAL UNIT 2.                                *
C     *    INPUT:     PRT - TO BE SCALED TO 1 AND WRITTEN        *
C     *                    TO LOGICAL UNIT 2.                    *
C     *               NPLOT - THE NUMBER OF THE PLOT             *
C     *               KEY - NUMBER OF CURVE THIS PLOT            *
C     ************************************************************
C


      SUBROUTINE STOW(PRT,NPTS)
      DIMENSION PRT(NPTS)
      COMMON XFM(256,2),CPHI(256,128),SPHI(256,128),PI,
     1IXT(10),IYT(10),KEY
      WRITE(3,13)NPTS
      WRITE(3,13)KEY
CKK   IF (KEY .NE. 1)GO TO 12
C     WRITE AXIS LABELS
      WRITE(3,10) (IXT(I),I=1,10)
      WRITE(3,10) (IYT(I),I=1,10)
10    FORMAT(10A4)
12    NPLOT =  NPLOT + 1
      BPRT = 0.0
13    FORMAT(I4)
      DO 100 I=1,NPTS
      IF (PRT(I) .GT. BPRT) BPRT = PRT(I)
100   CONTINUE
      IF (BPRT .LT. .00001) BPRT = 1.
      DO 200 I=1,NPTS
      PRT(I) = PRT(I) / BPRT
      T = FLOAT(I)
      WRITE(3,20)T,PRT(I)
20    FORMAT(2(3X,F9.5))
200   CONTINUE
      RETURN
      END
```

```
C
C
C   ************************************************
C   * THE HOLD SUBROUTINE TAKES THE INPUT FILE     *
C   * <FILIN> AND STORES IT IN THE OUTPUT FILE     *
C   * <FILOUT> FOR TEMPORARY STORAGE. THE FILE     *
C   * <FILIN> REMAINS UNCHANGED.                   *
C   ************************************************
C

      SUBROUTINE HOLD(FILIN,FILOUT,NPTS)
      DIMENSION FILIN(NPTS),FILOUT(NPTS)

      DO 100 I = 1,NPTS
      FILOUT(I)  = FILIN(I)
100   CONTINUE
      RETURN
      END

C
C   *******************************************************
C   * THE ALTER SUBROUTINE WILL ALTER THE COMMON ARRAY    *
C   * <WFM> AS SPECIFIED BY THE INPUT VARIABLES           *
C   * <SCALE> AND <SHIFT>.  THE ALTERED WFM IS OUTPUT     *
C   * IN THE VECTOR <ALT>.                                *
C   *******************************************************
C
C

      SUBROUTINE ALTER(ALT,WFM,SCALE,SHIFT,NPTS)
      DIMENSION ALT(NPTS),WFM(NPTS),TOLD(256),TNEW(256)
      COMMON XFM(256,2),CPHI(256,128),SPHI(256,128),PI,
     1IXT(10),IYT(10),KEY
      DO 100 I=1,NPTS
      TOLD(I) = FLOAT(I)
      TNEW(I) = TOLD(I) / SCALE + SHIFT
100   CONTINUE
      CALL INTP3(WFM,TOLD,ALT,TNEW,NPTS)
      RETURN
      END
```

```
C
C    ******************************************************************
C    *   INTP3 IS A SECOND ORDER INTERPOLATION BASED ON A            *
C    *   POLINOMIAL EQUATING TO FIRST AND SECOND DERIVATIVES         *
C    *   APPROXIMATED BY CENTRAL DIFFERENCES.  THE INPUT             *
C    *   VECTOR <XO> HAS OLD SAMPLES AT TIMES IN <TO>. THE           *
C    *   NEW SAMPLE TIMES ARE INPUT THROUGH ARRAY <TN> AND           *
C    *   THE COMPUTED SAMPLES AT THESE TIMES ARE OUTPUT IN           *
C    *   ARE OUTPUT IN THE VECTOR <XN> AND <WFM>.                    *
C    ******************************************************************
C
C
      SUBROUTINE INTP3(XO,TO,XN,TN,NPTS)
      DIMENSION X3(3),XO(NPTS),TO(NPTS),XN(NPTS),TN(NPTS)
      COMMON XFM(256,2),CPHI(256,128),SPHI(256,128),PI,
     1IXT(10),IYT(10),KEY
C
C
C     CHOSE THE <TO> SAMPLE TIME CLOSES TO THE WARPED
C     TIME HELD IN <TN>
C
      DO 60 I=1,NPTS
      DT = 100
      XPTS = FLOAT(NPTS)
      IF((TN(I) .GE. 1.) .AND. (TN(I) .LE. XPTS)) GO TO 5
      XN(I) = 0.0
      GO TO 60
5     DO 10 J=1,NPTS
      DTABS = ABS(DT)
      DIST = TO(J) - TN(I)
      DIST = ABS(DIST)
      IF (DTABS .LT. DIST) GO TO 25
      DT = TN(I) - TO(J)
10    CONTINUE
25    JTIME = J - 1
      J1 = -1
      DO 30 J=1,3
      J1 = J + JTIME-1
      IF ((J1 .GE. 1) .AND. (J1 .LE. NPTS))GO TO 29
      X3(J) = 0.0
      GO TO 30
29    X3(J) = XO(J1)
30    CONTINUE
      TIMN = TN(I)
      TIMO = TO(JTIME)
      XN(I) = FCN(X3,TIMO,TIMN)
60    CONTINUE
      DO 500 I=1,NPTS
500   CONTINUE
      RETURN
      END
```

```
C
C
C     FCN IS THE INTERPOLATION RULE.
C
      FUNCTION FCN(X,TO,TN)
      DIMENSION X(3)
C
C     COMPUTE THE COEFFICIENTS.
C
      A = (X(3) - 2.* X(2) + X(1)) / 2.
      B = (X(3) - X(1)) / 2.0 - 2.* A * TO
      C = X(2) - A * TO**2. - B * TO
C
C     COMPUTE THE INTERPOLATED VALUE.
C
      FCN = A * (TN**2) + B * TN + C
      RETURN
      END
```

```
C
C     ****************************************************
C     * THIS SUBROUTINE IS THE RESULT OF A CLOSED        *
C     * FORM CALCULATION OF THE CONTINUOUS SINC**2       *
C     * BEING TAKEN FROM THE TIME DOMAIN THROUGH THE     *
C     * ENTIRE FOURIER-MELLIN PROCESSING.  THE ALGORITHM *
C     * IS USED TO VERIFY THE FDM PROCESS.  THE OUTPUT   *
C     * FEATURE SPACE SHOULD BE IDENTICLE TO THAT PRODUCED *
C     * BY ANY OF THE MELLIN ALGORITHMS.  FOR THIS REASON *
C     * THE SAMPLE POINTS ARE SYNCHRONIZED WITH THOSE USED *
C     * ABOVE.  <NCOEF> FM COEFFICIENTS ARE USED.  THE   *
C     * COMMON VARIABLE <KEY> MUST BE SET TO 99 PRIOR TO *
C     * CALLING TO GET THE SINC**2 OUTPUT FEATURE SPACE. *
C     * TO OUTPUT THE CLOSED FORM SOLUTION FOR A RAMP IN *
C     * THE FREQUENCY DOMAIN, <KEY> MUST EQUAL 100.      *
C     ****************************************************
C
C


      SUBROUTINE CFORM(CF,NCOEF)

      DIMENSION CF(NCOEF)

      COMMON XFM(256,2),CPHI(256,128),SPHI(256,128),PI,
     1IXT(10),IYT(10),KEY

      IF(KEY .EQ. 100) GO TO 200
      IF(KEY .NE.99) RETURN
      DO 100 M=1,NCOEF
      XM = FLOAT(M) - 1.0
      HMAG = 1 + (PI * XM / 18.)**2
      CF(M) = SQRT(HMAG) / HMAG
100   CONTINUE
      GO TO 101
200   DO 202 I = 1,NCOEF
      XI = FLOAT(I) -1.0
      OMEGA = PI * XI / 18.
      CF(I) = 2./SQRT(4. + OMEGA**2)
202   CONTINUE

101   CONTINUE

      READ(2,20)KEY
      READ(2,10) (IXT(I),I=1,10)
      READ(2,10) (IYT(I),I=1,10)
10    FORMAT(10A4)
20    FORMAT(I4)

      CALL STOW(CF,NCOEF)

      RETURN
      END
```

```
C
C     ************************************************
C     * TITLE TITLES THE Y AXIS FOR THE PLOTTING     *
C     * PROGRAM ACCORDING TO THE ALGORITHM USED TO   *
C     * GENERATE THE FEATURE SPACE.                  *
C     ************************************************

      SUBROUTINE TITLE (ID)
      DIMENSION ID(5)
      COMMON XFM(256,2),CPHI(256,128),SPHI(256,128),PI,
     1IXT(10),IYT(10),KEY
      DO 100 I=1,5
      IYT(I) = ID(I)
100   CONTINUE
      RETURN
      END
```

LIST OF REFERENCES

1. Duda, R.O. and Hart, P.E., Pattern Classification and Scene Analysis, John Wiley and Sons, 1973.

2. Brigham, E.O., The Fast Fourier Transform, Prentice-Hall, Inc, 1974.

3. Oppenheim, A. and Schafer, R., Digital Signal Processing, Prentice-Hall, Inc., 1975.

4. Casasent, D. and Psaltis, D., "Position, Rotation, and Scale Invariant Optical Correlation," Applied Optics, vol.15, no. 7, July 1976.

5. Casasent, D., and Psaltis, D., "New Optical Transformations for Pattern Recognition," Proceedings of the IEEE, vol. 65, no. 1, January 1972.

6. Casasent, D. and Psaltis, D., "Scale Invariant Optical Correlation Using Mellin Transforms," Optics Communications, vol 17, no. 1, April 1976.

7. Casasent, D. and Szczutkowski, C., "Optical Mellin Transforms Using Computer Generated Holograms," Optics Communications, vol.19, no. 2, November 1976.

8. Altes, R. A., "The Fourier-Mellin Transform and Mammalian Hearing," J. Acoust. Soc. Am., vol. 63, no. 1, January 1978.

9. McNeil, O., "Digital Implementation of the Mellin Transform," Unpublished Paper for Radar Systems Branch, Naval Weapons Center, China Lake, Ca (code 3158)

10. Zwicke, P. E. and Kiss, I., "Invariant Feature Extraction for Target Classification," unpublished paper to be submitted to IEEE, United Technologies Research Center, 1981.

11. Sneddon, I. H., The Use of Integral Transforms, McGraw-Hill, Inc. 1972.

12. Abramowitz, M. and Stegun, I. A., Handbook of Mathematical Functions with Formulas, Graphs, and Tables, Dover Publications, Inc., 1964.

13. Naylor, D., "On a Mellin Type Integral Transform," Journ. of Math. and Mech., vol. 12, no. 2, 1963.

14. Hamming, R. W., Digital Filters, Prentice-Hall, Inc. 1977.

15. Oberhettinger, F., Tables of Mellin Transforms, Springer-Verlag 1974.

16. Moses, H. E., and Quesada, A. F., "The Power Series of the Mellin Transform with Applications to Scaling of Physical Quantities," Journal of Mathematics and Physics, vol. 15, no. 6, June 1974.

17. Helms, H. D., "Power Spectra Obtained from Exponentially Increased Spacings of Sample Positions and Frequencies," IEEE Transactions on Acoustics, Speech, and Signal Processing, vol. ASSP-24, no. 1, February 1976.

18. Shannon, M. and Steinig, W., Data Base Target Profile Modeling, paper presented at the Combat Identifications Conference, US Army Electronics Research and Development Command, Ft. Monmouth, N.J., 28 October 1981.

19. Carnahan, B. and Wilkes, J. O., Digital Computation and Numerical Methods, John Wiley and Sons, Inc., 1973.

20. Beyer, W. H., Standard Math Tables, CRC Press, Inc., 1978.

21. McDonnel, M., "A Clarification on the use of the Mellin Transform in Optical Pattern Recognition," Optics Communications, vol. 3, no. 3, June 1978.

22. Parker, S. R., "Discrete Signal Processing and Digital Filters," notes pending publishing, Copyright 1975.

23. Prost, R. and Goutte, R., "Linear Systems Identified by Mellin Deconvolution," International Journal of Control, vol. 23, no. 5, 1976.

24. Prost, R. and Goutte, R., "Performance of the Method of Linear Systems Identification, by Mellin Deconvolution," International Journal of Control, vol. 25, no. 1, 1977.

25. Gregory, R. L., Eye and Brain, The Psychology of Seeing, McGraw-Hill Book Company, 1973.

26. Haber, R. N., Information Processing Approaches to Visual Perception, Holt, Rinehart, and Winston, Inc, 1969.

27. Frisby, J. P., Seeing, Illusion, Brain, and Mind, Oxford University Press, 1980.

28. Carterette, E. C. and Friedman, M. P., ed., Handbook of Perception, Volume V, Seeing, Academic Press, 1975.

29. Oppenheim, A. V. and Lim, J. S., "The Importance of Phase in Signals," Proceedings of the IEEE, vol. 69, no. 5, May 1981.

30. Pearlman, W. A. and Gray, R. M., "Source Coding of the Descrete Fourier Transform," IEEE Transactions on Information Theory, vol. IT-24, September 1971.

31. Kermish, D., "Image Reconstruction from Phase Information Only, Journal of the Optical Society of America, vol. 60, no. 1, January 1970.

32. Skolnik, M. I., Introduction to Radar Systems, McGraw-Hill, Inc. 1980.

33. Hamming, R. W., Coding and Information Theory, Prentice-Hall, Inc., 1980.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Technical Information Center Cameron Station Alexandria, Virginia 22314	2
2.	Superintendent (Attn: Code 0142) Naval Postgraduate School Monterey, California 93940	2
3.	Superintendent (Attn: Code 62Wi) Naval Postgraduate School Monterey, California 93940	6
4.	Chief (Attn: Code 210) Office of Naval Research 800 N. Quincy St. Arlington, Virginia 22217	2
5.	Director (Attn: Code 00) Joint Cruise Missile Project Office Washington, D.C. 20360	1
6.	Director (Attn: Code 531) Joint Cruise Missile Project Office Washington, D.C. 20360	1
7.	Director (Attn: Code 5312) Joint Cruise Missile Project Office Washington, D.C. 20360	1
8.	Commander (Attn: Code 62D1) Naval Sea Systems Command Washington, D.C. 20362	1

9. Commander 2
(Attn: Code 315)
Naval Weapons Center
China Lake, California 93555

10. Commander 1
(Attn: Code PMA 258)
Naval Air Systems Command
Washington, D.C. 20631

11. Commanding Officer 1
(Attn: LCDR N. E. Huston)
Underwater Demolition Team 22
Naval Amphibious Base, Little Creek
Norfolk, Virginia 23521

12. Dr. P. E. Zwicke 1
United Technologies Research Center
East Hartford, Connecticut 06108

13. Navy Headquarters 1
(Attn: LCDR I.K.Chang)
Personnel and Education Department
Seoul, Korea

Thesis
H95755 Huston
c.1 Shift and scale invariant preprocessor.